ESCUELA TÉCNICA Y SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE CARTAGENA

PROYECTO FIN DE CARRERA

Diseño, implementación y simulación del procesador externo de sonido de un implante coclear

Autor: Miguel Fernández del Barrio
Director: Vicente Garcerán Hernández

Cartagena, Septiembre de 2013

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Capítulo 1

Introducción

1.1. Implantes

En general, un implante es una estructura de material biocompatible (generalmente titanio o siliconas) y dispositivos electrónicos que sirve para suplir carencias y malos funcionamientos orgánicos.

Desde los implantes dentales hasta los marcapasos, los implantes cocleares, los sistemas inteligentes de administración farmacológica o los sistemas electrónicos para paliar los efectos del Parkinson, los implantes llevan años en la vida del ser humano haciendo que enfermedades antes intratables ahora lo sean y mejoren la calidad de vida de los pacientes.

El implante coclear, una prótesis auditiva cuya función es la de sustituir la función de la coclea, ha supuesto un cambio muy importante en las soluciones técnicas para la discapacidad auditiva severa o profunda.

Se aplica actualmente sobre todo en poblaciones de personas con discapacidad auditiva adquirida y en niños con sordera prelocutiva en edad preescolar; sus resultados en dichos grupos se estiman altamente satisfactorios.

Un implante coclear puede ser definido como un aparato que transforma los sonidos y ruidos del medio ambiente en energía eléctrica capaz de actuar sobre las aferencias del nervio coclear, desencadenando una sensación auditiva en el individuo.

El concepto de estimulación eléctrica para producir sensaciones auditivas en el paciente con una hipoacusia profunda [1] no es nuevo. Volta, en 1800, colocó unas varillas de metal en sus dos oídos y las conectó a una fuente eléctrica. Aparentemente, antes de perder el conocimiento, oyó un sonido parecido al burbujeo del agua. A lo largo del siglo XIX y primera mitad del XX otros autores como Politzer, Ritter, Gradenigo, Andreef, Gersuni, Volokhov, Jones, Stevens y Lurie, efectuaron experiencias algo más sofisticadas aplicando corriente alterna a electrodos ubicados en las proximidades del oído obteniendo así sensaciones auditivas en los pacientes.

El primer implante coclear fue realizado por A. Djurno y C. Eyries en Francia en 1957. Ellos insertaron un único hilo de cobre dentro de la coclea de un varón de 50 años totalmente sordo, logrando éste percibir el ritmo del lenguaje. En 1961 W. House realizó sus dos primeras implantaciones colocando un electrodo de oro en la escala timpánica. Posteriormente en 1968, llevó a cabo otros implantes, empleando esta vez un sistema de seis electrodos que había sido elaborado por su colaborador J. Urban. El éxito obtenido en estos casos constituyó un empuje decisivo para el desarrollo de los implantes cocleares.

Otros grupos en San Francisco (Schindler, Merzenich y Michaelson), Francia (Chouard), Alemania (Banfai) y Austria (Burian), iniciaron protocolos clínicos con implantes cocleares en la década de los 70. En Australia, G.M. Clark, de la Universidad de Melbourne, comenzó en 1967 una serie de trabajos de investigación sobre la fisiología de la estimulación eléctrica del nervio coclear en animales. En 1978 y 1979 practicó sus dos primeras implantaciones con un prototipo multicanal intracoclear, obteniendo resultados altamente esperanzadores.

Actualmente, después de una experiencia que supera los 40.000 implantes cocleares en el mundo, se puede considerar esta técnica como no experimental, habiendo quedado demostrada su eficacia en el tratamiento de la hipoacusia profunda.

Los implantes cocleares pueden clasificarse atendiendo a tres criterios: ubicación de los electrodos (intra o extracocleares), número de canales (mono o multicanales) y forma de tratar la señal sonora (extracción o no de los distintos formantes del sonido).

También los implantes cocleares pueden ser clasificados de acuerdo al tipo de electrodos (monopolares, bipolares), método de estimulación (pulsátil, continua) o forma de transmisión de las señales a nivel de la piel (conexiones percutáneas o transcutáneas).

Todos estos sistemas tienen ventajas e inconvenientes, pero ha quedado demostrado que la estimulación multicanal/intracoclear produce una superior capacidad de comprensión de la palabra hablada que la estimulación monocanal o extracoclear.

Los implantes cocleares están indicados en pacientes que presentan una hipoacusia neurosensorial bilateral profunda de asiento coclear, que se benefician de forma insuficiente o nula de los audífonos y que además se sienten motivados hacia un implante coclear. Partiendo de los criterios de la "Federal Food and Drug Administration", esta indicación se concreta en individuos con umbrales auditivos bilaterales superiores a 90dB de media en las frecuencias de 500Hz, 1kHz y 2 kHz que además presentan, en campo libre con la utilización de audífonos, unos umbrales superiores a 55 dB y una discriminación de la palabra inferior al 40 %, empleando listas abiertas de palabras. No obstante, es previsible que en un futuro próximo se introduzcan cambios en estos criterios audiométricos, de forma que en determinados casos de hipoacusias neurosensoriales severas grado 2, con mínimos beneficios de los audífonos, sea indicado un implante coclear.

En la actualidad se consideran contraindicaciones las siguientes situaciones:

1. Malformaciones congénitas que cursan con una agenesia bilateral de la coclea.

2. Ausencia de funcionalidad de la vía auditiva o presencia de enfermedades que originen una hipoacusia de tipo central.

3. Enfermedades psiquiátricas severas.

4. Enfermedades que contraindiquen la cirugía bajo anestesia general.

5. Ausencia de motivación hacia la implantación.

6. No cumplimiento de los criterios audiológicos.

En este Proyecto de Fin de Carrera se desarrolla una parte del sistema externo de un implante coclear. En primer lugar se realizará un modelado software que posteriormente será exportado a un sistema hardware. El objetivo es la comprobación y comparación de los resultados respecto a los datos obtenidos de otras distintas estrategias.

En primer lugar y de forma introductoria, en la presente memoria se expondrán los órganos que intervienen en el oído y las patologías que trata de corregir el implante coclear. Desde el punto de vista meramente estructural un implante coclear se percibiría como un transductor mecánico-eléctrico seguido de una estructura en paralelo de procesado de señal. En esta se podrían identificar muchos componentes como filtros, rectificadores etc muy utilizados en el mundo de la electrónica. Sin embargo este conocimiento sólo va ser de utilidad a la hora de implementar modelos electrónicos una vez comprendida la complejidad del implante coclear y cuyo entendimiento requiere de una fase inicial de documentación apoyándose en estudios del área biológica. Se debe tener en cuenta que el modelo electrónico del implante está intentando sustituir siempre un funcionamiento orgánico de base bioquímica mediante el uso de la electrónica. Por consiguiente, la comprensión tanto de los órganos implicados en el sentido del oído como sus patologías ayudarán a tener una amplia perspectiva de los objetivos de este proyecto.

1.2. El oído

El oído es uno de los cinco sentidos del ser humano. Su buen funcionamiento permite relacionarnos entre nosotros mediante comunicación verbal así como el aprendizaje y captación de señales auditivas provenientes del entorno.

El oído comprende distintos órganos que se encargan de la audición y del equilibrio. En términos médicos se le conoce como órgano vestibulococlear. Dicho órgano puede dividirse en tres etapas:

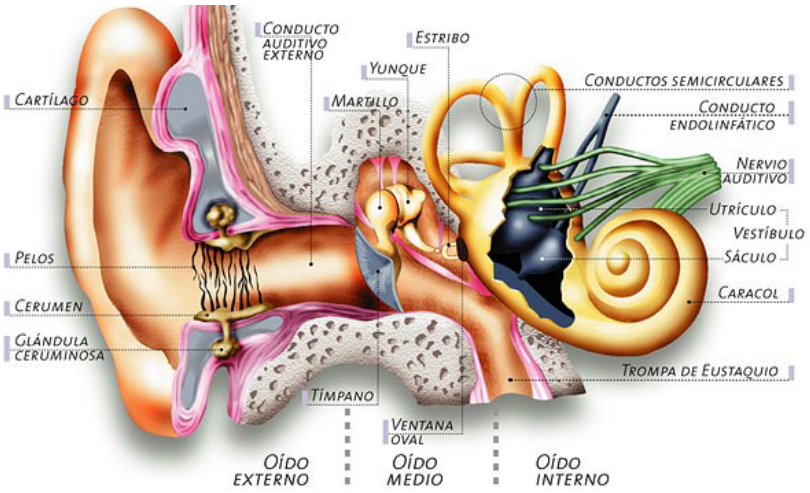


Figura 1.1: El oído

- Oído externo, formado por el pabellón auditivo y el conducto auditivo externo.

- Oído medio, formado por la cavidad timpánica, la membrana timpánica, los osteocillos óticos (huesecillos del oído), senos y celdas mastoideos, así como la tuba faríngea o faringotimpánica (antes denominada Trompa de Eustaquio).

- Oído interno o laberinto se encuentra dentro del hueso temporal. Puede dividirse morfológicamente en laberinto óseo y laberinto membranoso. El laberinto óseo es la cápsula ósea que rodea al laberinto membranoso, y éste último consiste en un sistema hueco que contiene a la endolinfa. Entre laberinto óseo y membranoso se encuentra la perilinfa, que es en parte un filtrado de la sangre y en parte difusión de líquido cefalorraquídeo. La endolinfa se produce en la estría vascular.

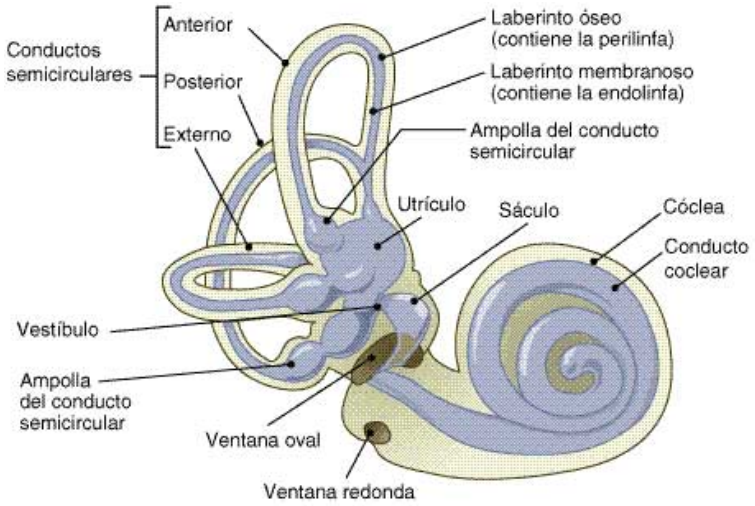


Figura 1.2: Conductos auditivos

El caracol o cóclea, contiene en su interior al Órgano de Corti, que es un mecanorreceptor. Está formado por células ciliadas que descansan sobre la membrana basilar [2]. Los cilios de estas células se encuentran en contacto con la membrana tectoria. Cuando se produce un estímulo, el estribo ejerce presión sobre la ventana oval, esto genera una onda en la perilinfa que viaja a lo largo de la cóclea desplazando la membrana basilar. Esto produce flexión de los cilios en contacto con la membrana tectoria lo que se traduce en cambios de potencial celular que generan estímulos nerviosos a través de las células bipolares del nervio coclear.

1.3. El Implante coclear para tratar casos de sordera profunda

La sordera profunda suele darse a edades tempranas y tiene su origen en malformaciones genéticas de la cóclea (por ejemplo el síndrome de Mondini[3]), o del nervio auditivo y también origen infeccioso, principalmente meningitis.

Mediante una tomografía se pueden observar deformaciones de la cóclea de origen genético. Suelen ser cócleas que no llegan a desarrollarse en la etapa fetal y dan lugar a órganos incompletos en forma de cisterna o cavidad comunicante.

Tanto en las cócleas osificadas por procesos infecciosos como en las mal formadas hay que proceder pensando que puede haber un número menor de células bipolares auditivas como resultado de una lesión por la infección o por restricción de neuronas al detenerse el desarrollo embrionario en una etapa muy temprana; por ende es imposible restablecer la función orgánica a la perfección pero los pacientes se pueden beneficiar ostensiblemente con el uso del implante coclear.

En la figura 1.3 se muestra una cóclea incompleta en forma de cisterna.

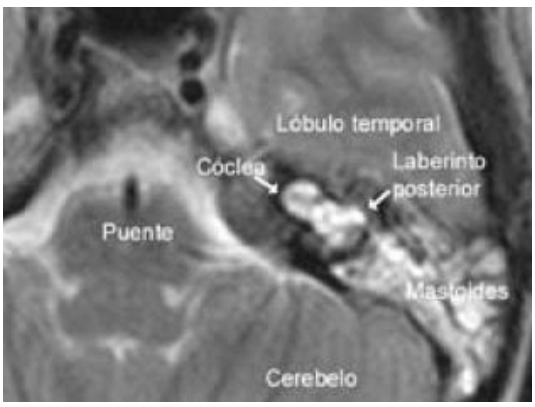


Figura 1.3: Cóclea defectuosa

Los implantes cocleares son sistemas electrónicos cuya finalidad es la de suplir este mal funcionamiento orgánico. Son sistemas multietapa que consiguen mimetizar con una determinada precisión el funcionamiento deficiente o nulo de una parte del sistema auditivo.

El implante auditivo consta básicamente de dos partes: una es externa al cuerpo del paciente y otra interna y requiere una implantación quirúrgica con anestesia general.

La parte externa consta de un sistema electrónico que suple todo el tejido y órganos que abarcan desde el oído externo hasta el oído interno (concretamente la cóclea) , haciendo la función de procesador de sonido y transmitiendo la información procesada a una bobina imantada que a su vez la transmitirá a la parte interna por radiofrecuencia.

La parte interna estaría formada por un sistema de recepción que incluye un dispositivo imantado que se coloca debajo de la piel y un conjunto de electrodos a los que accede mediante una perforación ósea en el hueso mastoides. En el caso de este proyecto son veintidós electrodos los que estimulan zonas concretas del nervio auditivo.

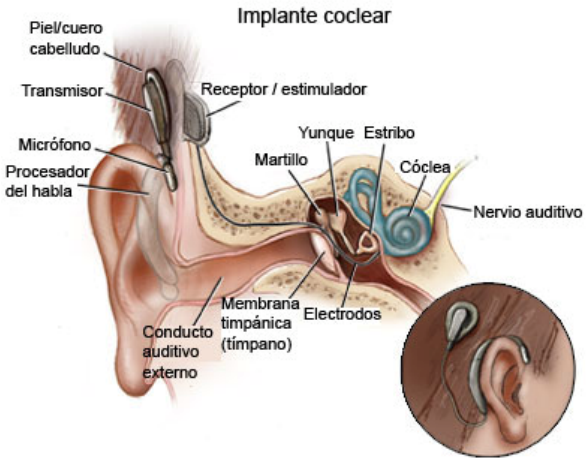


Figura 1.4: Implante coclear

Los implantes cocleares son una opción para personas con una sordera profunda las cuales no presentan mejoría con el uso de audífonos. La audición mediante el dispositivo implantado no es perfecta, es decir que la persona implantada no escucha igual los sonidos que una persona sana. Es un sustituto que intenta paliar el defecto orgánico y permitir al cerebro procesar la información básica de una fuente sonora.

Un especialista médico (concretamente el otorrinolaringólogo con ayuda de especialistas en el ámbito de los implantes) es el que determina si el paciente es apto para la operación. Los principales criterios para determinar esta aptitud del paciente son los siguientes:

-Pacientes mayores de un año de edad con sordera profunda que no presenta mejoría con el uso de audífonos y con capacidad de adaptación y adherencia al uso y cuidado del dispositivo. El paciente ha de estar motivado ya que es un proceso lento y con resultados en el medio plazo.

-Pacientes tolerantes a la anestesia y que hayan sido vacunados contra la meningitis. Es un requisito indispensable del protocolo médico.

-El paciente debe tener acceso a un sistema de logopedia y tutoría del implante con el fin de ayudarle a obtener óptimos resultados. Determinados pacientes necesitarán también ayuda psicológica.

-Las expectativas del paciente han de estar acotadas, deben ser realistas. Con la tecnología actual aún no es posible restablecer la función auditiva de un individuo sano dada la naturaleza discreta del implante y restricciones propias de sistemas electrónicos actuales.

Durante la década de los noventa, se especulaba sobre posibles daños producidos por la estimulación eléctrica en la corteza cerebral y el nervio auditivo a edades tempranas, sin embargo estudios de la Agencia de Alimentos y Medicamentos de Estados Unidos (FDA, Food and Drug Administration) han demostrado que esto no es así. Esto ha dado lugar a que la implantación pueda darse hoy en día en pacientes a partir de un año de edad [4].

A edades tempranas de implantación se observan mejores resultados. Esto es debido a la plasticidad de la vía auditiva; la plasticidad es la habilidad de las conexiones neurales a ser modificadas por crecimiento de las sinapsis como efecto de la transmisión de estímulos adecuados. Es rápidamente restringida en niveles superiores de la vía a mayor edad. Este proceso de plasticidad neuronal comienza a limitarse rápidamente a partir de los 6 años de vida por lo cual hasta esa edad se considera que es el periodo crítico para el implante coclear y para el desarrollo de la vía auditiva post implante.

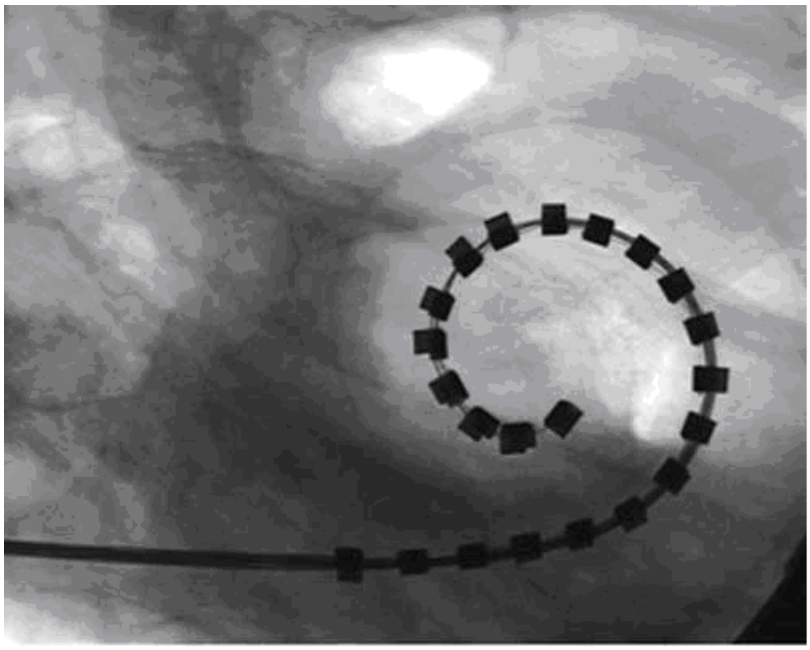


Figura 1.5: Electrodos situados en la cóclea

La operación para implantar el dispositivo implica ciertos riesgos, aunque la probabilidad de los mismos ha disminuido mucho a lo largo de los años en que se ha evolucionado en esta técnica. Además la fase preoperatoria va orientada a evitarlos en lo posible mediante el uso de resonancias magnéticas, tomografías y distintas técnicas de visualización y análisis.

Capítulo 2

Uso de FPGAs en el diseño externo de un implante

Con la evolución de los sistemas de integración a gran escala VLSI (Very Large Scale Integration), cada vez se está perfeccionando más la técnica de los implantes. Concretamente con la utilización de elementos lógicos programables FPGA (Field Programmable Gate Array) se consigue una versatilidad y reducción de costes sin precedentes respecto al uso de los circuitos integrados de aplicación específica o ASICs (Application Specific Integrated Circuits) y una velocidad de procesado muy superior a la ofrecida por modelados software.

Hoy en día las computadoras tienen procesadores multinúcleo y son capaces de gestionar varios hilos de procesado concurrentemente. A pesar de esto, a la hora de trabajar con aplicaciones que requieren alta velocidad de procesado, es necesario usar sistemas que no estén limitados por la arquitectura VonNeuman. La capacidad inherente de procesado en paralelo de la FPGA usada en este Proyecto Fin de Carrera otorga ventajas notables a la implementación del sistema.

La principal ventaja de usar sistemas software frente a hardware radica en el hecho de que las herramientas de programación (C,C++, Java), son más sencillas que las de diseño hardware (VHDL) al ofrecer un nivel de abstracción superior. Sin embargo la inclusión de herramientas con interfaces más amigables, como es el caso de System Generator de Xilinx, y las ventajas que ofrece un diseño hardware lo hacen superior y preferible a un modelado software.

2.1. Ventajas del uso de FPGAs frente a otras estrategias

Para entender las ventajas que ofrece una FPGA es necesario conocer su funcionamiento y en qué se basan. Las FPGA son dispositivos semiconductores inventados por la empresa Xilinx en 1984. Su estructura y diseño permite que estos sean reprogramados por el usuario. Esto ofrece posibilidades muy extensas de diseño que anteriormente sólo eran posibles mediante el uso de CPLDs (Complex Programmable Logic Device) que presentan una menor

integración de componentes, una estructura más rígida y carencias en cuanto a bloques aritméticos complejos, como multiplicadores embebidos en el propio dispositivo. Si medimos la densidad de los elementos lógicos programables en puertas lógicas equivalentes (número de puertas NAND equivalentes que podríamos programar en un dispositivo) podríamos decir que en un CPLD hallaríamos del orden de decenas de miles de puertas lógicas equivalentes y en una FPGA del orden de cientos de miles hasta millones de ellas.

Las principales ventajas que hacen preferible el uso de la tecnología FPGA frente a otros sistemas son las siguientes:

-Alto rendimiento en procesamiento en paralelo. Gracias a las FPGA se consigue romper el paradigma de la ejecución secuencial y se logra más capacidad de cómputo por ciclo de reloj. En el modelado de bandas críticas (banco de filtros y demás componentes configurados en paralelo) usado en este proyecto, las capacidades de procesado en paralelo hacen más eficiente el cómputo en tiempo real y a alta velocidad.

-Ahorro de tiempo. Esto se debe a que una vez implementado el diseño este puede ser portado a una placa FPGA de forma inmediata. Las placas FPGA se encuentran en el mercado con una disponibilidad muy alta. Esto evita el lento y arduo proceso de fabricación que sucede al diseño.

-Precio ventajoso. Si estimamos un ratio entre la capacidad de cómputo y unidades monetarias, las FPGAs son ventajosas frente a sistemas de implementación más rígidos. Además a largo plazo suponen un ahorro debido que, aunque cambien los requerimientos del diseño, podrá seguir siendo implementado en la FPGA disponible (siempre y cuando fuese elegida previamente con unos criterios de sobredimensionamiento) sin necesidad de invertir tiempo y dinero en un rediseño hardware. Para la fabricación de un número pequeño de unidades de un sistema la ventaja económica se hace muy notable. El abaratamiento del diseño hardware debido a la economía de escala (fabricación de dispositivos en gran número que deriva en un ahorro de costes) recae sobre el fabricante, en nuestro caso es Xilinx.

-Mayor fiabilidad al no presentar los problemas típicos de otros sistemas con mayores niveles de abstracción. Al gestionar todos los recursos mediante herramientas proporcionadas por el fabricante se evitan colisiones debidas a gestión con sistemas operativos, drivers etc. El hardware de la FPGA será destinado a tareas concretas y operará en paralelo con otros sistemas.

-Mayor integración con equipos hardware y herramientas software. Al tener un ecosistema de aplicaciones (tanto del fabricante como de terceros) se abre un mundo de posibilidades a la hora de exportar datos para almacenarlos, procesarlos, representarlos etc.

En la figura 2.1 podemos observar un bloque matricial simplificado de la estructura de una FPGA.

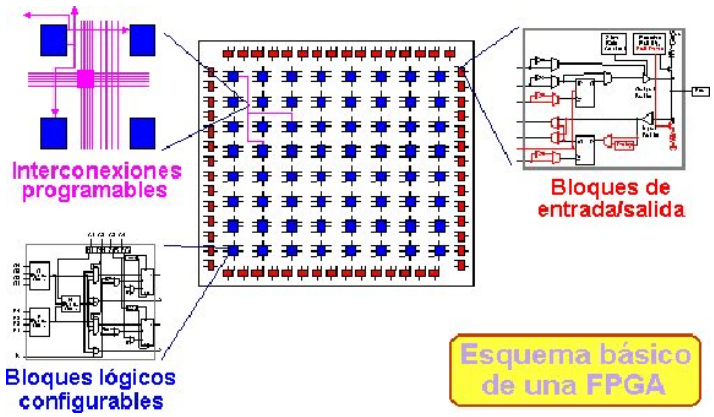


Figura 2.1: Esquema Básico FPGA

La figura 2.2 muestra el layout de nivel superior de una FPGA.

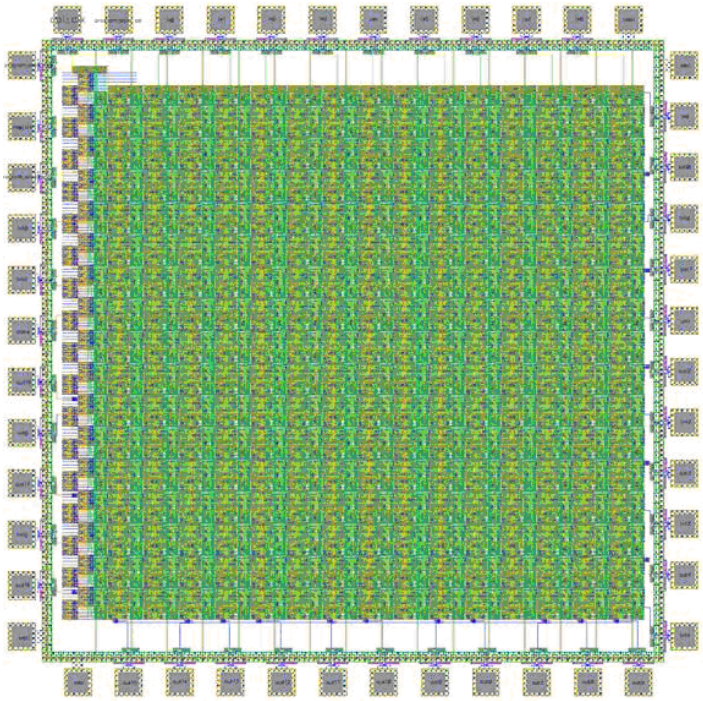


Figura 2.2: Layout FPGA

La figura 2.3 muestra la evolución paralela de los procesadores -6X y la familia Xilinx de FPGAs. Se observa una disminución de las dimensiones de los transistores y un crecimiento exponencial de su número.

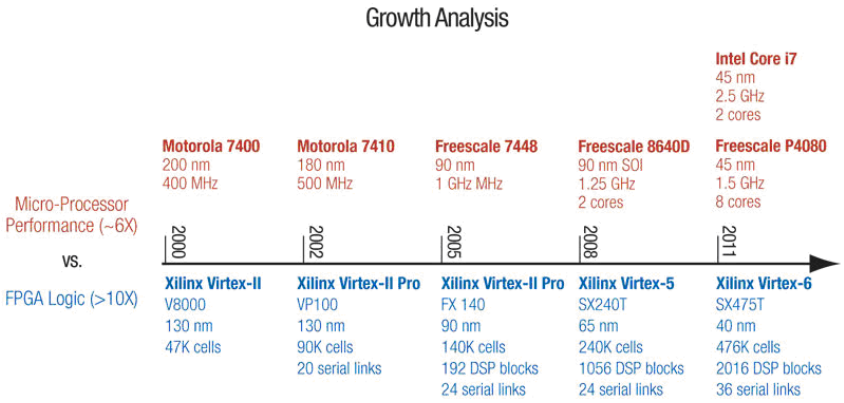


Figura 2.3: Evolución de la tecnología FPGA

En la etapa inicial del diseño del sistema se desconoce el modelo exacto de FPGA a usar. Será cuando se haya completado la fase de diseño y prueba por computador cuando, usando las herramientas ofrecidas por la ToolBox System Generator, seleccionemos el modelo de FPGA acorde con los requerimientos hardware. Se intentará, dentro de unos límites, sobredimensionar el sistema FPGA para poder incrementar sus prestaciones futuras de ser necesario. El diseño que será implementado en tecnología FPGA corresponde al primer bloque de la figura 2.4 (la cual muestra el diagrama de un implante completo) excluyendo el procesador de sonido.

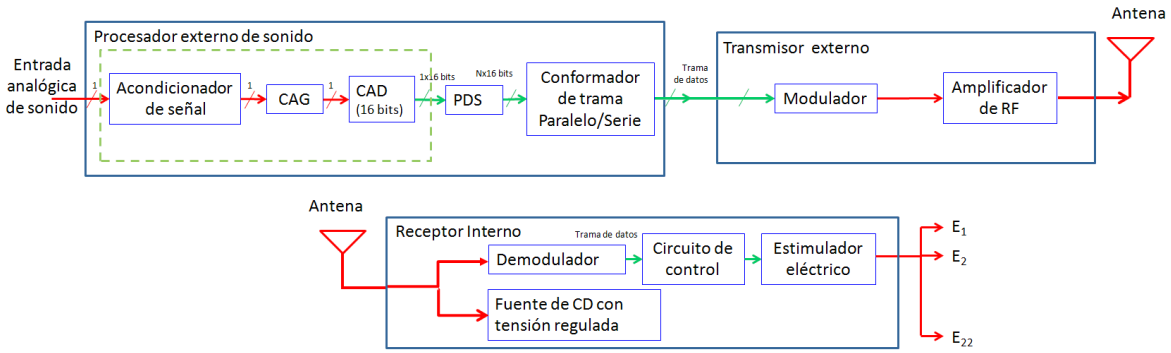


Figura 2.4: Diagrama de bloques del implante coclear

Capítulo 3

Implementación del DSP (Digital SOUND PROCESSOR)

Introducción

La herramienta que se usará en este proyecto para la implementación del DSP es System Generator, que es una ToolBox para Simlulink propietaria de Xilinx. Consiste en una herramienta de alto nivel para el diseño de DSPs que permite la implementación de sistemas de procesamiento en paralelo con los últimos modelos de FPGAs del mercado. La elección de ésta herramienta se debe a la versatilidad y facilidad de uso al incorporar un generador de código VHDL a partir de diseños esquemáticos de alto nivel. Esto hace que la tarea de diseño sea más visual e intuitiva. Además incorpora multitud de bloques prediseñados (Xilinx DSP Targeted Design Platform) y herramientas de comunicación así como herramientas para el cálculo de potencia y recursos necesarios del diseño.

Las estrategias de codificación del habla se basan en el principio neurofisiológico de "disparo estocástico neuronal", basado en el modelo estocástico de redes neuronales [5, capítulo I]. La estimulación neuronal a tasas bajas induce un sincronismo de disparo que no representa el modelo biológico por esto surgen estrategias de codificación como CIS, una de las más usadas comercialmente en los implantes cocleares.

En este proyecto se abarca la estrategia CIS (Continuous Interleaved Sampling). CIS es una estrategia de procesamiento del habla para los implantes cocleares en la que se presentan breves pulsos en cada electrodo en una secuencia sin solapamiento, en el caso de este proyecto con una tasa de 2000 pps (pulsos por segundo).

La secuencia de proceso de la estrategia CIS se muestra en el diagrama de la figura 3.1

Figura 3.1: Estrategia CIS

Otra estrategia usada en los implantes cocleares es la SAS (Simultaneous Analog Stimulation). En esta estrategia la fase de compresión de la señal, a diferencia de la estrategia CIS, se da al inicio del proceso. En esta estrategia se envía a cada electrodo la onda procesada, pudiéndose estimular electrodos adyacentes. La desventaja de esta estrategia es que es mas susceptible a interferencias entre canales.

Podemos encontrar multitud de estudios y comparativas entre ambas estrategias. El hecho de que una u otra de mejores resultados dependerá de la posición de los electrodos en la cóclea. En [6] podemos encontrar un estudio comparativo entre estas dos estrategias. En éste se probaron ambas estrategias en diferentes pacientes y se determinó que el orden de uso de una u otra estrategia no afecta a la puntuación que se otorgó al paciente.

En un primer lugar un análisis estadístico de las puntuaciones otorgadas a cada estrategia reveló que aunque la estrategia CIS tenía mayores puntuaciones respecto a SAS, el análisis de varianza no pudo determinar esta conclusión.

La figura 3.2 muestra una comparativa del porcentaje de éxito entre estrategias CIS y SAS bifásicas y monofásicas.

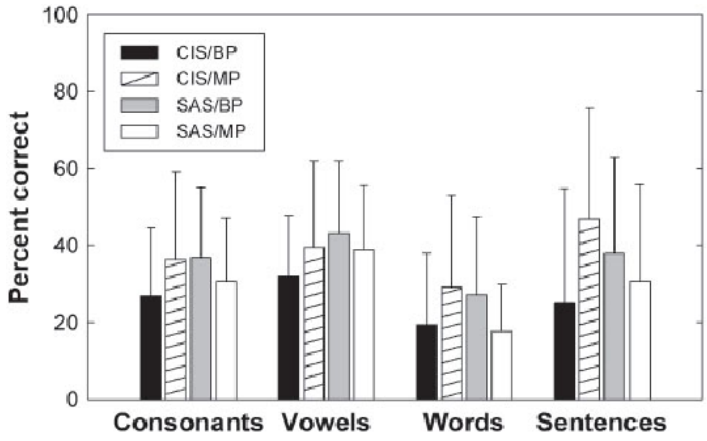


Figura 3.2: Comparativa entre CIS y SAS

En las figuras 3.3 y 3.4 se muestran los porcentajes de éxito para el caso de consonantes, vocales, monosilábicos CNC (consonant-nucleus-consonant) y sentencias HINT (Hearing In Noise Test).

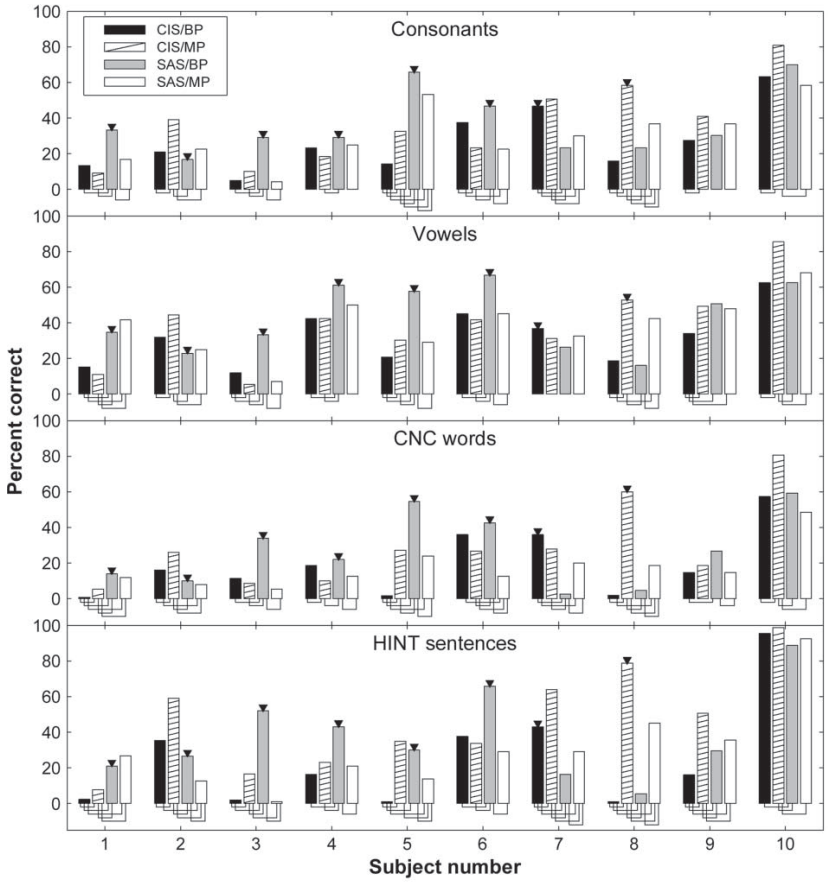


Figura 3.3: Porcentaje de éxito

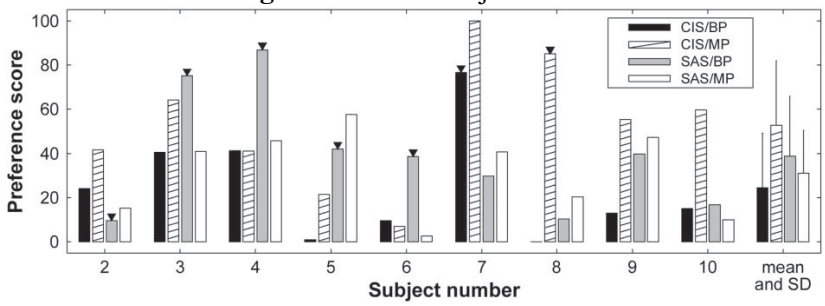


Figura 3.4: Comparativa individualizada

CNC es un test que consiste en la audición por el implantado de dos listas de cincuenta palabras monosilábicas de la forma consonante-vocal-consonante. La puntuación de este test equivale al número de aciertos sobre las cien palabras escuchadas.

HINT consiste en la audición por el paciente de diferentes frases en entorno ruidoso (con la fuente de ruido en diferentes ángulos de emisión) y en entorno libre de ruido y su posterior repetición. Al final del test se establece el ratio SNR (Signal- Noise Rate) para que la repetición de frases sea exitosa en un 50 % de los casos. La puntuación de este test se basa en éxito y fracaso de la repetición de las frases escuchadas, partiendo del estudio rea-

lizado a una población de cincuenta individuos que presentan una capacidad auditiva normal.

El sistema DSP comprende una fase previa a la modulación de pulsos bifásicos dentro del algoritmo CIS (Continous Interleaved Sampling). En la figura 3.5 se muestra el diagrama de bloques a implementar del DSP.

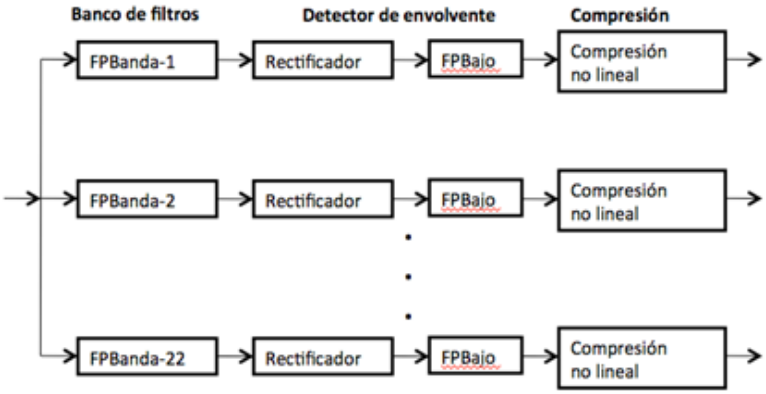


Figura 3.5: Diagrama de bloques a implementar del DSP.

El diseño del anterior esquema se realizará de forma secuencial, comprobando que cada sistema funciona de forma individual ante entradas distintas y que también funciona correctamente una vez se han integrado todos los componentes que le preceden.

3.1. Banco de filtros pasa-banda

El banco de filtros paso banda corresponde a la primera etapa del modelado del sistema de análisis frecuencial que es la cóclea. Dicho comportamiento puede definirse mediante dos características:

-Existen varios receptores auditivos que procesan la componente espectral de la señal sonora

-Cada uno de los receptores procesa diversas componentes espectrales.

3.1.1. La escala Bark

El oído humano tiene características que son no lineales como la sensibilidad y la selectividad auditiva. Los modelos que consideran estas variables no lineales se aproximan más a la realidad que los modelos puramente lineales donde solo se contemplan el nivel de presión y filtrados pasa-banda de tercios de octava. En este proyecto se usa la escala Bark [7] a la hora de realizar los veintidós filtros pasa banda. La escala Bark es una escala psicoacústica,

relaciona las frecuencias absolutas en Hertz con frecuencias preceptuales. El ancho de banda de los filtros usados en esta escala crece de forma no lineal con su frecuencia central [8].

Estudios de la discriminación en frecuencia del oído han demostrado que en las bajas frecuencias, tonos con unos cuantos Hertz de separación pueden ser distinguidos; sin embargo, en las altas frecuencias para poder discriminar los tonos se necesita que estén separados por cientos de Hertz. En cualquier caso, el oído responde al estimulo más fuerte que se presente en sus diferentes regiones de frecuencia; a este comportamiento se le da el nombre de bandas críticas. Los estudios muestran que las bandas críticas son mucho más estrechas en las bajas frecuencias que en las altas; el 75 % de las bandas críticas están por debajo de los 5 kHz, lo que implica que el oído recibe más información en las bajas que en las altas frecuencias. Las bandas críticas tienen un ancho de aproximadamente 100 Hz para las frecuencias de 20 a 400 Hz; este ancho aumenta de manera logarítmica a medida que aumenta la frecuencia.

Cada uno de los filtros pasa-banda implementados corresponde a una banda crítica. Las bandas críticas están relacionadas con el fenómeno psicoacústico de enmascaramiento. Dicho fenómeno explica cómo la presencia de un ruido en una banda crítica afecta al proceso auditivo. El sistema auditivo no es capaz de diferenciar dos sonidos presentes en la misma banda crítica o en bandas adyacentes solapadas. Si la amplitud de un sonido en esta banda no es capaz de superar el umbral producido por el ruido que se introduce en dicha banda (enmascarador), este sonido quedará enmascarado.

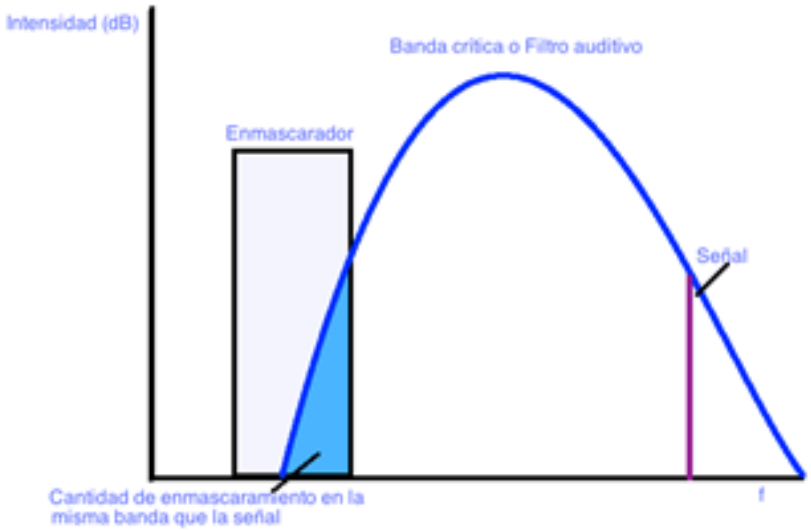


Figura 3.6: Enmascarador.

Así pues, se define una banda crítica (BC) como un intervalo de frecuencia que representa la máxima resolución frecuencial del sistema auditivo en diversos experimentos psicoacústicos. Adicionalmente, puede decirse que una BC constituye el intervalo de frecuencia en el cual el oído interno efectúa una integración espacial (es decir, espectral) de la intensidad de

la señal sonora: la BC es el intervalo en el cual se "suma"la energía de las distintas componentes espectrales de la señal. Cada una de estas bandas críticas se relaciona unívocamente con una zona determinada de la membrana basilar y abarcan un mismo número de células ciliadas.

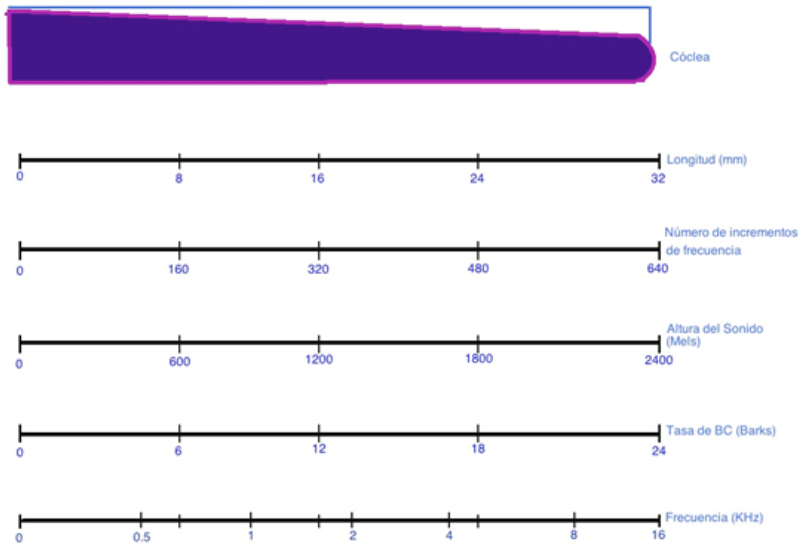


Figura 3.7: Comparativa entre escalas

Usando la escala Bark es posible conocer la zona de estímulo en la membrana basilar dependiendo de la frecuencia. La siguiente ecuación determina la razón de banda crítica "z" medida en Barks respecto a la frecuencia en kHz.

$$z = 13\ arctan(0{,}76f) + 3{,}5\ arctan\left(\frac{f}{7{,}5}\right)^2$$

La figura 3.8 es una ilustración de la cóclea y la membrana basilar. Se observa que conforme nos alejamos de la ventana oval nos encontramos en zonas donde se detectan las longitudes de onda más altas.

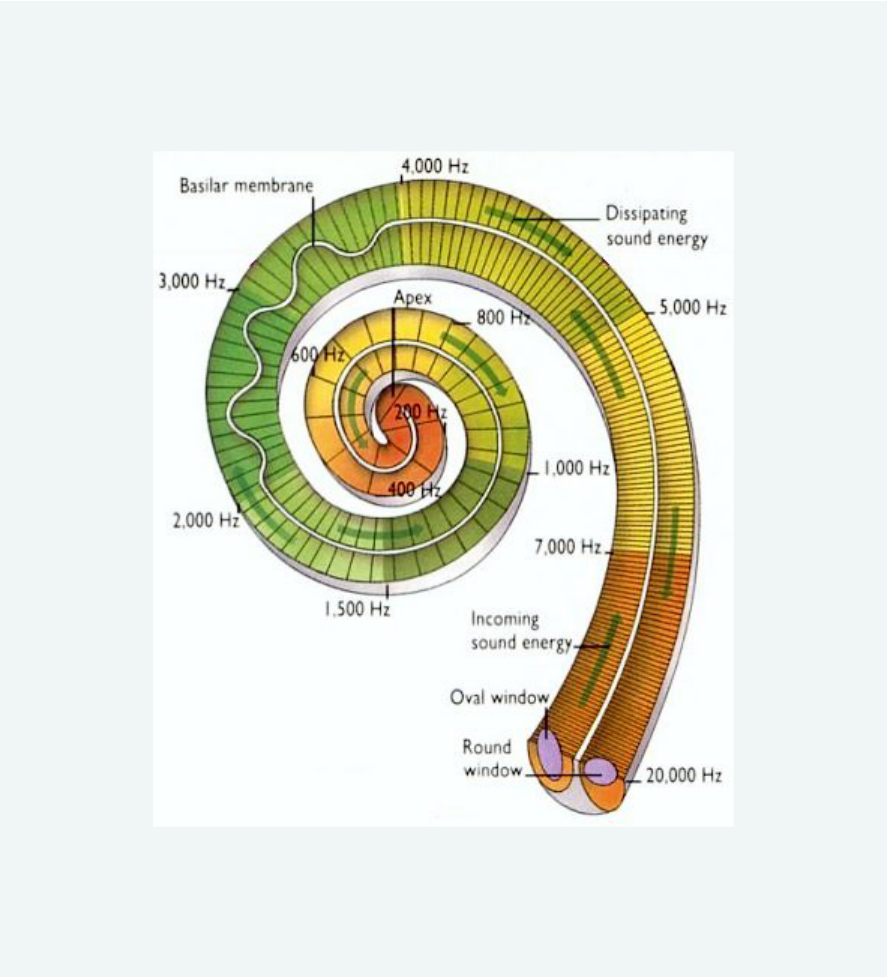


Figura 3.8: Zonas de estímulo en la cóclea según las frecuencias

Los valores enteros que adquiere z corresponden con las frecuencias de corte de los filtros cuando usamos el modelo de escala Bark. El ancho de banda de los filtros medido en Barks es constante e igual a la unidad. El ancho de banda en Hz puede calcularse con la siguiente fórmula, en función de la frecuencia central de cada banda crítica, en kHz.

$$CB_c = 25 + 75\ \left[1 + 1{,}4\ f_c^2\right]^{0{,}69}$$

En la figura 3.9 se representa la función de ancho de banda CBc frente a una linealización de la misma y una recta constante de 100 Hz.

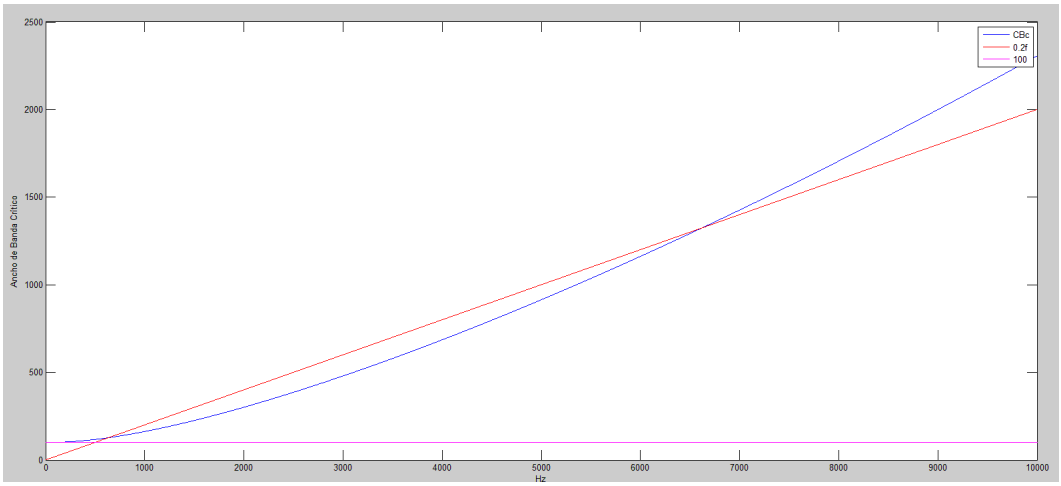


Figura 3.9: Función ancho de banda

En la figura 3.10, que es una ampliación de la anterior, se puede observar como para frecuencias inferiores a 500Hz el ancho de banda crítico es prácticamente constante, de 100Hz.

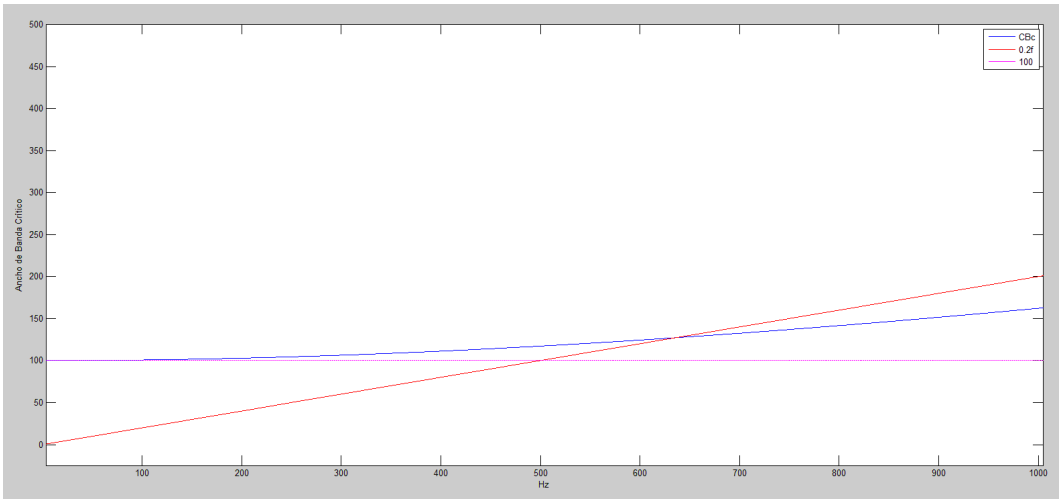


Figura 3.10: Función ancho de banda ampliada

3.1.2. Estrategia de diseño de filtros

Los filtros digitales

Los filtros digitales realizan cálculos directamente sobre las señales muestreadas y presentan numerosas ventajas respecto a los analógicos. Dichas ventajas son las siguientes:

-Se eliminan problemas característicos de sistemas analógicos como tolerancias o derivas de componentes del circuito. Se evitan comportamientos no ideales de resistencias, condensadores o amplificadores. Tampoco existen problemas de impedancia de entrada ni salida, ni efectos de adaptación de impedancias entre etapas. Todos los parámetros del filtro se encuentran digitalizados y no presentan derivas temporales.

-Versatilidad en su implementación: pueden ser implementados fácilmente con multitud de herramientas de desarrollo y se pueden modelar por software o en hardware.

-Intervalo dinámico más amplio ya que, a diferencia de los modelos analógicos, los modelos digitales permiten variar dicho intervalo mediante la modificación del número de bits de la secuencia (para el límite superior del intervalo) y mediante el proceso de cuantificación y los redondeos para el límite inferior (ruido de cuantificación y errores de redondeo). En los modelos analógicos estas limitaciones vienen dadas por la alimentación (saturación de la salida) y por derivas que limitan el límite inferior, que no siempre pueden modificarse.

-Respuesta dinámica: El ancho de banda del filtro digital está limitado por la frecuencia de muestreo, mientras que en dispositivos electrónicos utilizados, los filtros analógicos con componentes activos suelen estar restringidos por los amplificadores operacionales.

-Conmutabilidad: Si los parámetros de un filtro se conservan en registros, los contenidos de dichos registros pueden ser modificados a voluntad. Además, estos filtros se pueden conmutar, pudiéndose multiplexar en el tiempo para procesar varias entradas a la vez.

FIR e IIR, dos estrategias de diseño

A la hora de realizar el diseño de filtros digitales debemos elegir entre dos estrategias distintas, dicha decisión dependerá del objetivo del filtro y de los parámetros con los que vamos a trabajar. Dependiendo de estos parámetros (necesidad de velocidad, control de estabilidad) realizaremos un filtro IIR o FIR.

Los filtros FIR presentan una respuesta finita al impulso. Son capaces de funcionar en un amplio rango de tasas de muestreo y son generalmente bien soportados por un gran número de herramientas y estrategias distintas de diseño.

Partiendo de la ecuación en diferencias que modela el comportamiento dinámico de estos sistemas:

$y_k = a_1 y_{k-1} + a_2 y_{k-2} + ... + a_n y_{k-n} + b_0 x_k + b_1 x_{k-1} + ... + b_m x_{k-m}$

Si los coeficientes a son cero, se trata de un sistema FIR de orden m, quedando la ecuación de la forma:

$y_k = +b_0 x_k + b_1 x_{k-1} + ... + b_m x_{k-m}$, siendo su ecuación de transferencia en dominio z:

$G(z) = b_0 + b_1 z^{-1} + ... + b_m z^{-m}$

Los filtros IIR presentan una respuesta infinita al impulso. Se usan en tasas de muestreo inferiores a 200Hz, son filtros recursivos, es decir que hay una realimentación para calcular la salida pues esta depende de la entrada.

$y_k = a_1 y_{k-1} + a_2 y_{k-2} + ... + a_n y_{k-n} + b_0 x_k + b_1 x_{k-1} + ... + b_m x_{k-m}$

siendo la función de transferencia en el dominio z:

$$G(z) = \frac{\sum_{k=0}^{M} b_k \cdot z^{-k}}{\sum_{k=0}^{N} a_k \cdot z^{-k}}$$

Las principales ventajas del filtrado IIR frente a FIR son la velocidad superior cuando el número de coeficientes del filtro FIR se hace elevado y el ahorro de componentes en su implementación ya que con un filtro IIR de orden inferior a un FIR se consiguen resultados igual de óptimos.

Por otro lado los filtros IIR presentan la desventaja de no ser siempre estables y además presentan el problema de cuantificación en aritmética fija; esto quiere decir que un filtro IIR puede ser estable (polos dentro del circulo unidad en el diagrama Pole-Zero) en aritmética flotante y no serlo en aritmética fija (se produce un desplazamiento en los polos en el proceso de cuantificación). Debido a esto será necesario comprobar la estabilidad de los filtros IIR implementados.

En este proyecto se usarán filtros IIR de orden 2 pues debido a las ventajas anteriormente citadas. Para la implementación del banco de filtros del proyecto, se va a usar filtrado IIR de tipo Butterworth. Las frecuencias del banco de filtros vienen determinadas en la siguiente tabla:

	$F_C(Hz)$	$F_L(Hz)$	$F_H(Hz)$
1	100.0	82.3	117.7
2	123.2	104.2	142.2
3	151.8	131.2	172.3
4	187.0	164.6	209.5
5	230.4	205.6	255.2
6	283.9	256.2	311.5
7	349.7	318.5	381.0
8	430.9	395.3	466.5
9	530.9	489.9	571.9
10	654.0	606.4	701.7
11	805.8	749.9	861.7
12	992.8	926.8	1058.7
13	1223.2	1144.8	61301.6
14	1507.0	1413.2	1600.7
15	1856.6	1744.0	1969.2
16	2287.4	2151.6	2423.3
17	2818.2	2653.7	2982.7
18	3472.1	3272.3	3672.0
19	4277.8	4034.4	4521.1
20	5270.4	4973.4	5567.3
21	6493.3	6130.3	6856.3
22	8000.0	7555.7	8444.4

3.1.3. FDATool en el diseño de filtros digitales

MATLAB proporciona una herramienta de diseño de filtros llamada FDAtool. Esta herramienta permite obtener los coeficientes del filtro así como el diagrama de polos-zeros, la implementación del filtro usando componentes básicos, etc.

Dicha herramienta nos permite calcular todos los coeficientes y parámetros característicos para exportarlos al entorno workspace e implementar los filtros en bloques SIMULINK. Una vez obtenidos realizamos una implementación con bloques de la TOOLBOX System Generator.

La interface gráfica de FDATool nos permite configurar las características del filtro, el tipo de filtro (FIR o IIR) y visualizar la respuesta esperada del filtro entre otras posibilidades.

La figura 3.11 muestra la pantalla principal de la interface gráfica de usuario

FDATool nos permite generar un bloque que modela el filtro especificado. Tenemos la opción de generarlo como una caja negra que al acceder a ella desplegará un menú de configuración o, como nos interesa en este caso, un modelado basado en bloques ya existentes de SIMULINK.

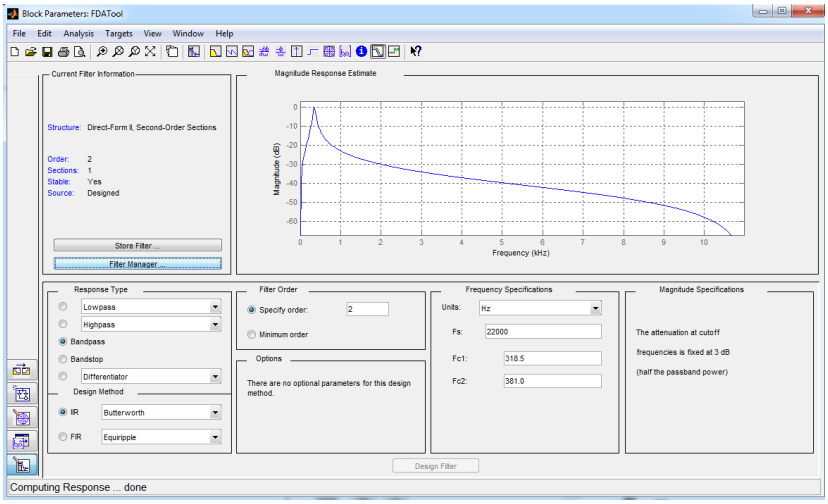


Figura 3.11: Interface gráfica FDAtool

La implementación del filtro que realiza FDATool es la forma directa II, que consiste en una estructura canónica donde el número de retardos introducidos es igual al orden del filtro. La ventaja de esta estructura es que minimiza el número de componentes hardware requeridos.

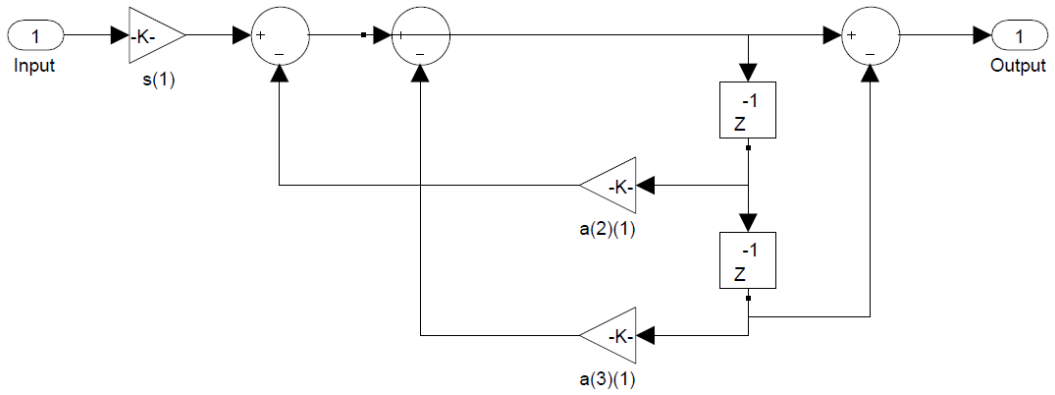


Figura 3.12: Diseño del filtro en Simulink

Si se requiriese el uso de filtros de orden superior, FDATool realizaría una configuración de filtros canónicos de orden 2 en cascada. Esto tiene como justificación evitar en lo posible los errores de cuantificación cuando el orden del filtro se hace muy grande.

La figura 3.13 muestra cómo implementa FDATool el filtro anterior con orden 4.:

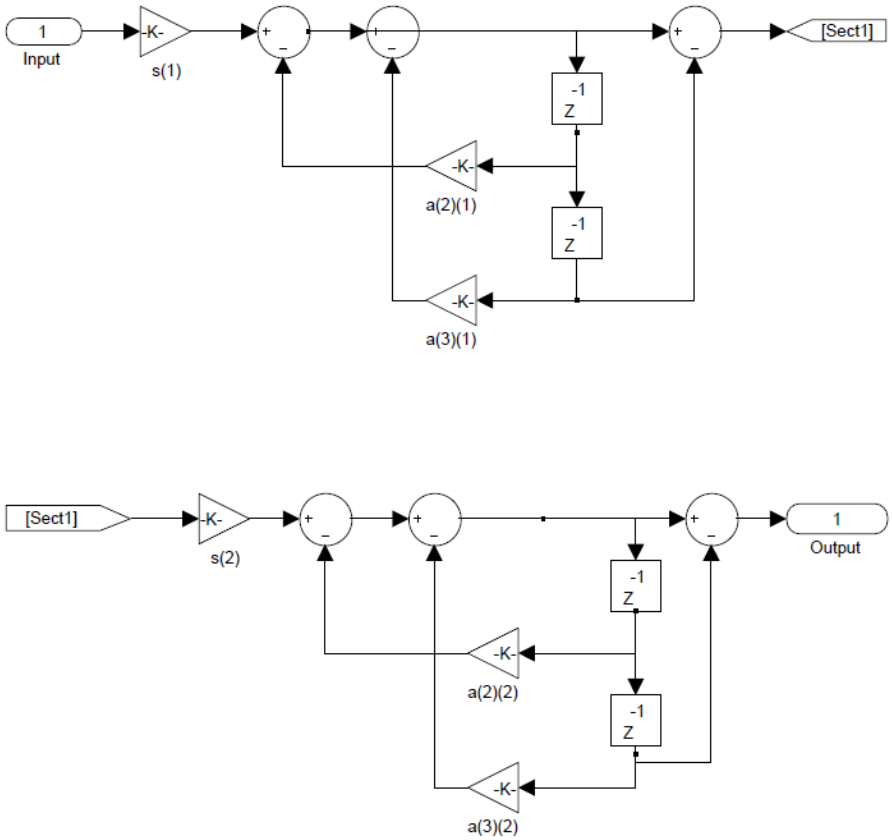


Figura 3.13: Filtro de orden 4 con FDATool

En la imagen de la figura 3.14 observamos la implementación análoga a la proporcionada por FDATool con los bloques de System Generator.

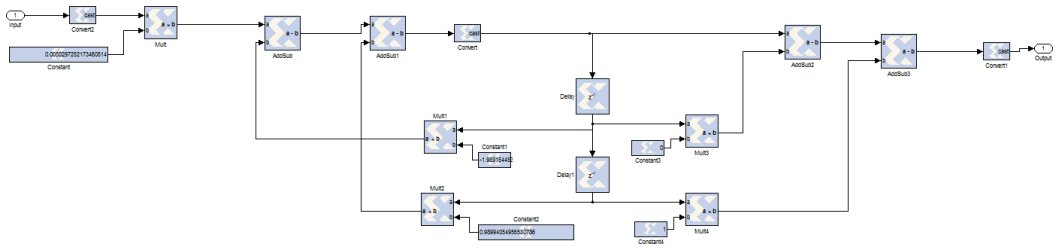


Figura 3.14: Filtro en System Generator

Todos los componentes del filtro funcionan con una salida " Output type Full ". Se generan las salidas con un número de bits acorde a lo requerido, sin establecer a priori ningún tipo

de restricción. Es importante establecer un número de bits suficiente en los bloques Constant para cargar los coeficientes del filtro. Estos deben ser del tipo Signed (2s Comp).

Una vez implementados todos los filtros paso banda en aritmética fija realizamos un test para verificar el buen funcionamiento de los mismos. En la figura 3.15 se muestra el esquemático usado para el test, aparecen las señales sinusoidales que se usarán para el test, los bloques de filtros y visualizadores de espectro.

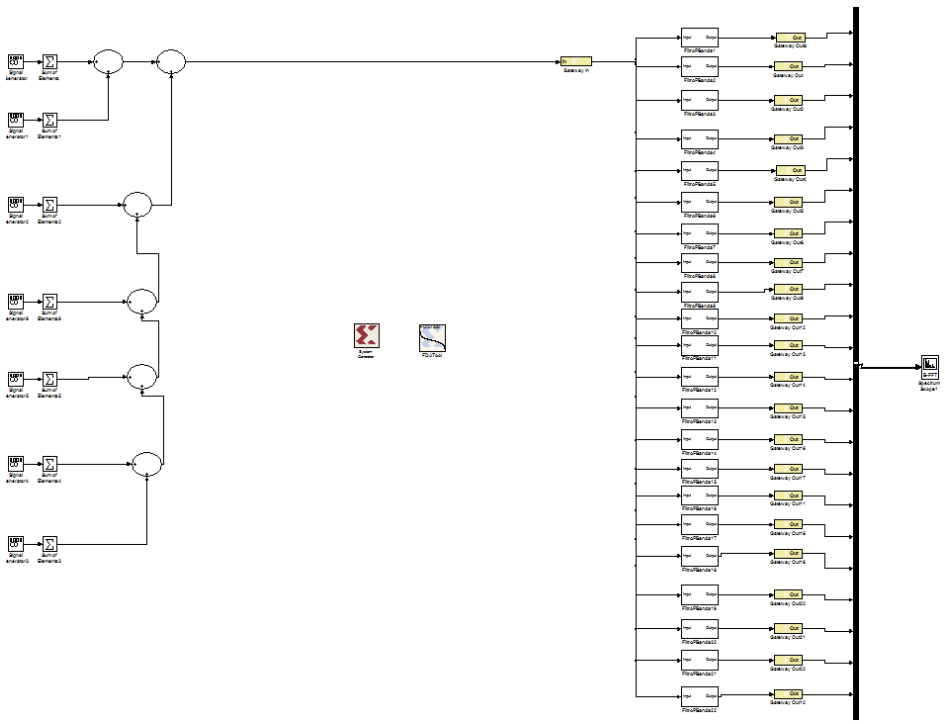


Figura 3.15: Esquema para la visualización del banco de filtros

Cada uno de los bloques Signal Generator (ANEXO 1) es capaz de generar un conjunto determinado de señales sinusoidales cuyas frecuencias se establecerán usando la sintaxis de MATLAB dentro del campo de configuración de dicho bloque. Además es necesario usar el bloque Sum of Elements (ANEXO 1) para obtener la adición del conjunto de sinusoidales generadas y evitarla un bus de datos por cada señal.

Mediante los bloques Spectrum Scope podemos observar la respuesta en frecuencia de cada uno de los filtros y comprobar que están implementados correctamente (Spectrum Scope del Anexo1).

La figura 3.16 muestra el espectrograma del conjunto del senoidales visualizado en un bloque Spectrum Scope. Se ha intentado obtener el mayor número de componentes frecuenciales entre 0 y 8500Hz a fin de obtener la respuesta en frecuencia de cada uno de los filtros.

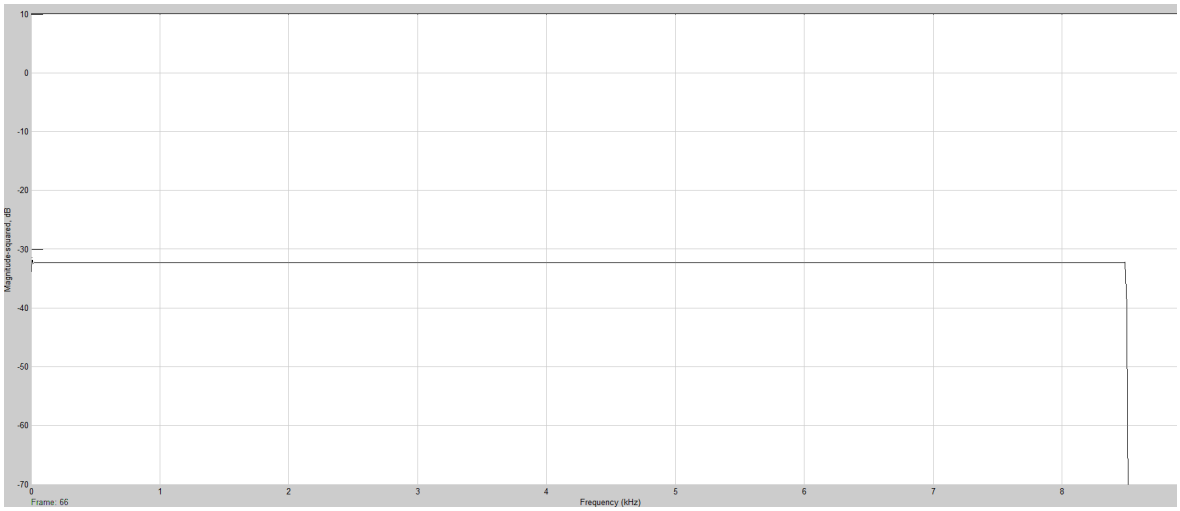


Figura 3.16: Espectrograma del conjunto de sinusoidales

A continuación (figura 3.17) se muestran las respuestas frecuenciales del banco de filtros visualizadas en un bloque Spectrum Scope

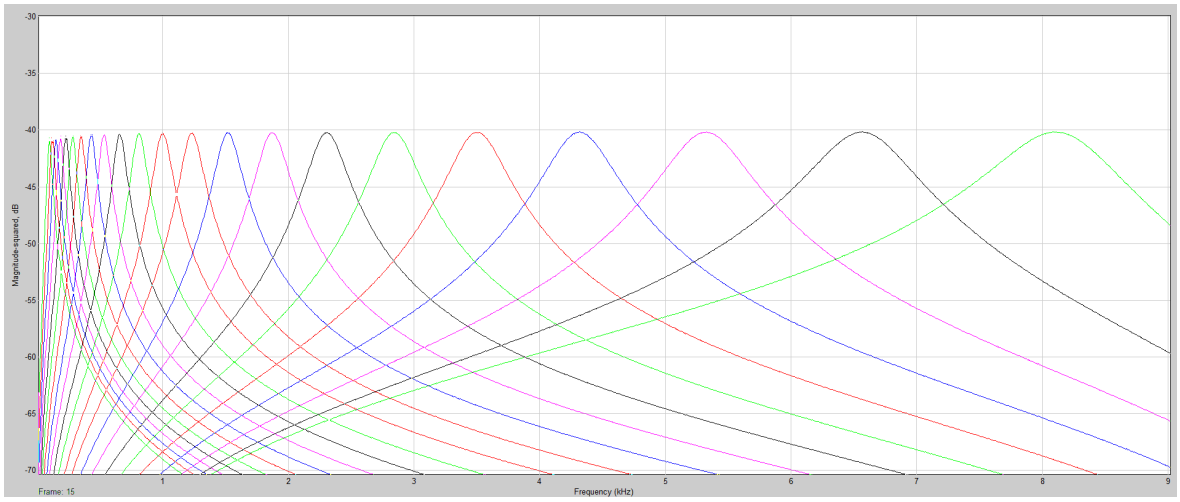


Figura 3.17: Respuestas frecuenciales

Los bloques Spectrum Scope presentan serias limitaciones cuando lo que se pretende es realizar un tratamiento de los datos obtenidos por los filtros. Es por esto que también se ha realizado un esquemático a fin de portar los datos obtenidos de cada filtrado al entorno de trabajo workspace de MATLAB.

En esta figura 3.18 se muestra el esquemático usado. La diferencia consiste en el uso de los bloques To Workspace.

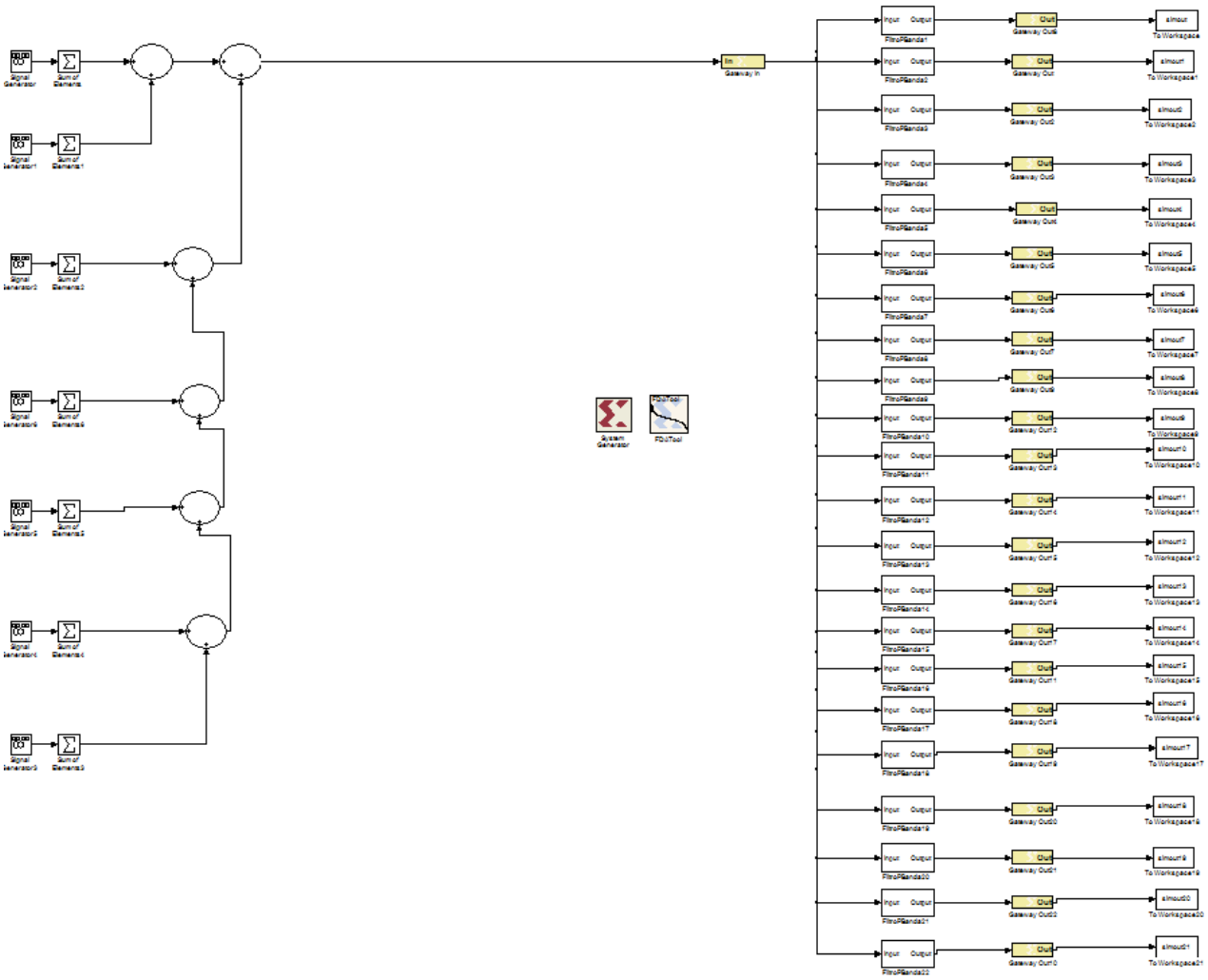


Figura 3.18: Datos de filtrado

Mediante los bloques To Workspace (ANEXO 1) podremos trasladar los datos de salida al entorno de trabajo de MATLAB. Los valores de las señales de salida muestreadas se almacenan en la variable simout.signals.values en forma de array de tipo double.

Haciendo uso de la función FFT proporcionada por MATLAB y la función de representación plot, podemos visualizar la densidad espectral de potencia de los filtros en la misma gráfica (ANEXO 2.4).

Realizamos un código como el anterior para cada una de las muestras obtenidas de cada filtro. En las figuras 3.19 y 3.20 podemos ver la representación del espectrograma del banco de filtros.

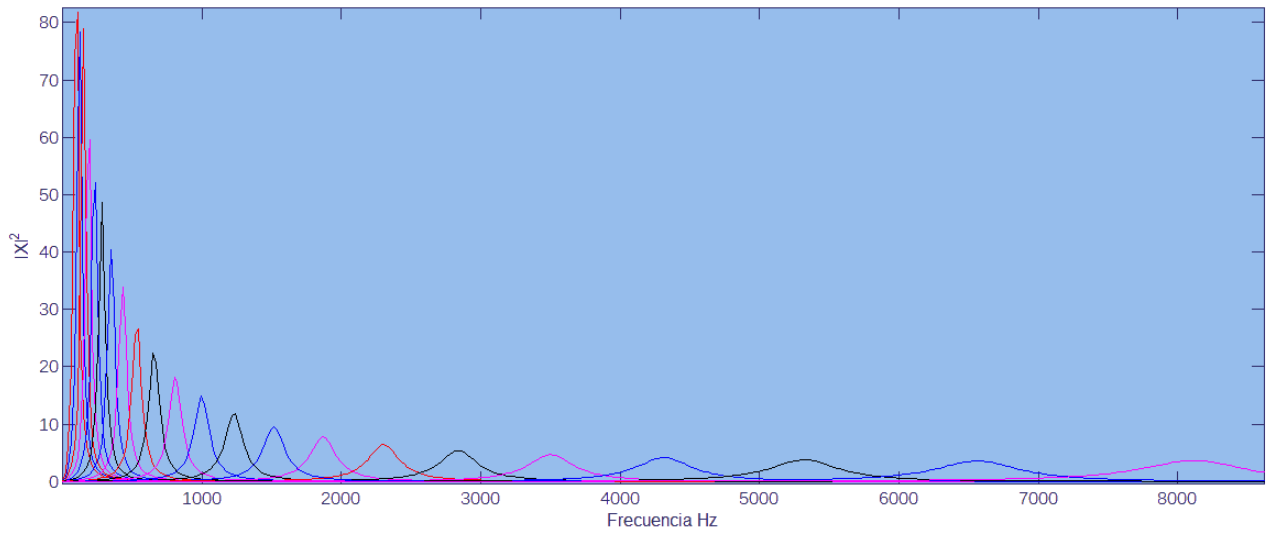


Figura 3.19: Densidad espectral de potencia del banco de filtros

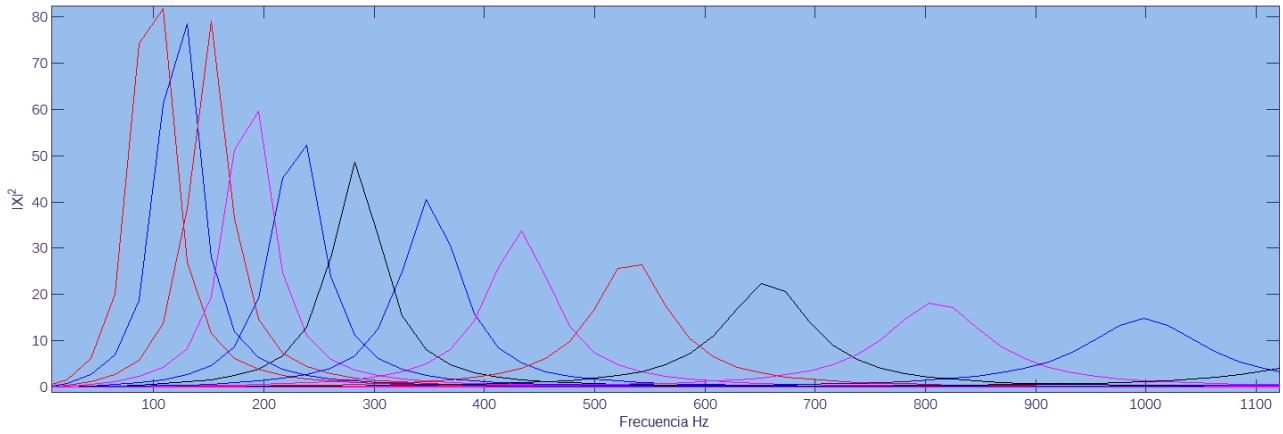


Figura 3.20: Zoom

En la anterior representación se observa que la densidad espectral de potencia es inferior conforme la frecuencia de corte del filtro crece. Esto se debe a que el ancho de banda de los filtros con mayor frecuencia de corte es superior y la potencia se distribuye entre un rango más amplio de componentes frecuenciales.

3.2. Detección de envolvente

La detección de envolvente se realiza mediante un rectificador y un filtrado paso-bajo. Dicho sistema extrae la energía de cada canal y la presenta como una señal de baja frecuencia a la salida del sistema.

3.2.1. Rectificador

La rectificación es un proceso que se da en todas las sinapsis corporales [9]. Su fundamento es un proceso bioquímico complejo que excede el alcance de este Proyecto Fin de Carrera. Este modelado de rectificación de media onda se apoya en resultados empíricos obtenidos por científicos del área biomédica. En "Mechanisms of synaptic transmission"se explica como el proceso de rectificación es producto de una segregación de sustancias llamadas neurotransmisores, como la noradrenalina y la dopamina, entre un sistema emisor y un sistema receptor.

En las sinapsis nerviosas no existe circuito cerrado formado por tejidos sino que es una transmisión química en un fluido. En los mamíferos, la información entre las neuronas se transmite a través de una sustancia química denominada neurotransmisor, que se libera en las sinapsis como respuesta a un estímulo específico. El neurotransmisor secretado actúa en sitios receptores especializados y altamente selectivos, que se localizan en la célula postsináptica, lo que provoca cambios en el metabolismo de ésta, los cuales modifican su actividad celular. La transmisión sináptica sólo se da en dirección ortodrómica, del emisor al receptor, y nunca en dirección antidrómica.

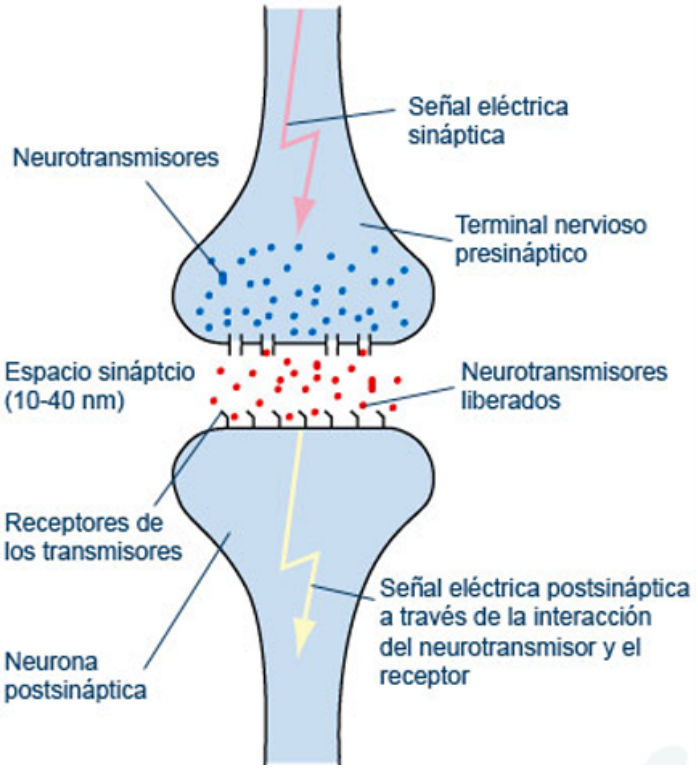


Figura 3.21: Transmisión sináptica

El objetivo es diseñar un bloque que realice una rectificación de media onda, es decir que

elimine las componentes negativas de la señal de entrada del filtrado pasa-banda precedente. El diseño es sencillo; se hace una interpretación de la señal (codificada en complemento a 2) como Unsigned y se compara dicha señal con su codificación invertida. Se pueden dar estos casos:

-Si la señal de entrada es negativa el bit más significativo será uno (debido a la codificación en complemento a dos) y será mayor que la señal invertida y provocará una salida 0 en el comparador. Esta salida elige d0 en el multiplexor, es decir una salida igual a 0.

-Si por el contrario la señal de entrada positiva, la señal invertida será mayor, lo que provocará una salida igual a 1 en el comparador y se seleccionará la salida d1 del multiplexor, que es la misma señal de entrada al bloque rectificador.

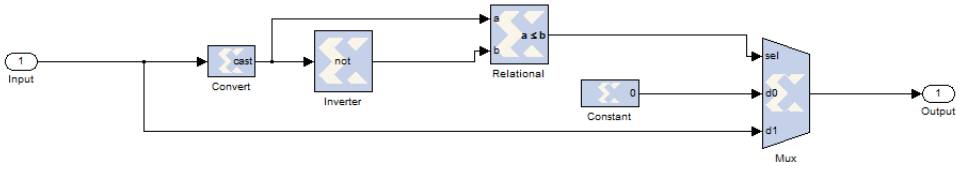


Figura 3.22: Rectificación de media onda

A continuación (figura 3.23) se observa la señal de entrada, la cual toma valores positivos y negativos. Esta señal usada para la prueba es fruto de la suma de múltiples señales sinusoidales después de sufrir un filtrado paso banda.

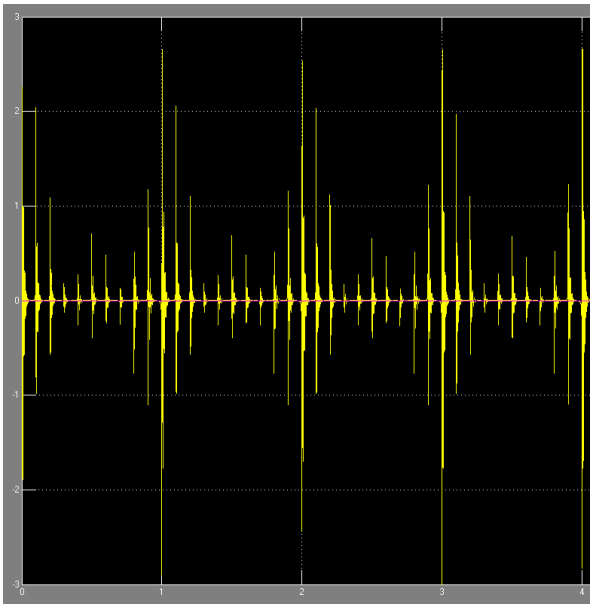


Figura 3.23: Señal de entrada al rectificador

La figura 3.24 muestra la señal resultante a la salida del rectificador. Se puede observar

como sigue manteniendo las mismas características que la señal original sin embargo se ha sesgado la parte negativa.

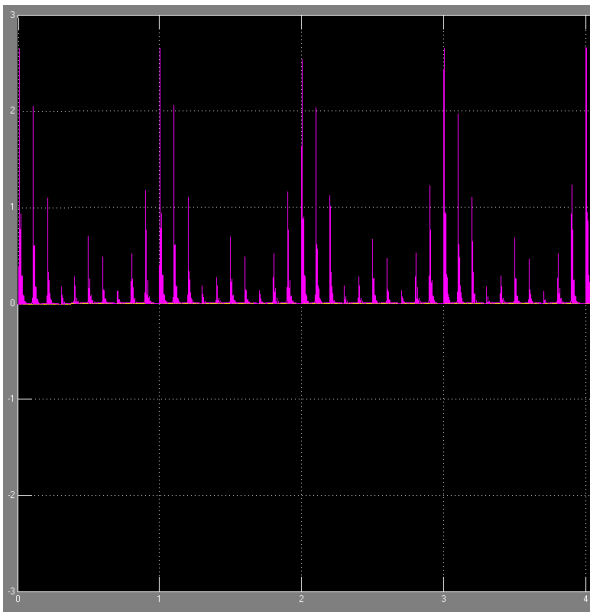


Figura 3.24: Señal a la salida del rectificador

3.2.2. Filtrado paso bajo

Corresponde al último paso para la realización de la detección de envolvente. Este filtro tiene como frecuencia de corte 200Hz y es de tipo Butterworth, de segundo orden. El método de implementación es similar al de los filtros paso banda.

La frecuencia de corte de 200Hz es la más comúnmente utilizada para implantes cocleares, aunque esto depende de la marca comercial. Algunas compañías implementan una versión modificada del algoritmo CIS denominada HiRes-CIS en la cual la frecuencia de corte de este filtro paso-bajo oscila entre 200Hz y 700Hz y se usan filtros FIR con transformadores de Hilbert para calcular la envolvente, evitando así la posible distorsión de fase de los filtros IIR.

3.3. Sistema de compresión

Este sistema tiene como fin convertir el amplio rango de valores eléctricos de entrada en un rango más preciso de estimulación eléctrica con fin auditivo.

El algoritmo de compresión es una función potencial conocida como power-law. La expresión de la misma es la siguiente:

$$y = Ax^p + B$$

A y B son parámetros que dependen de magnitudes eléctricas y que serán distintos dependiendo del paciente que va a ser intervenido. Las expresiones de A y B son las siguientes:

$$A = \frac{MCL - THL}{x_{max} - x_{min}}$$

$$B = THL - Ax_{min}$$

Donde X_{max} y X_{min} son los valores máximo y mínimo del rango dinámico de la señal de entrada. MCL es el máximo nivel audible que permanece en la zona de comodidad auditiva y THL es el mínimo nivel audible.

La realización de una función potencial mediante el uso de la ToolBox de Xilinx ofrecida para Simulink (MATLAB) y, en general, para sistemas eléctricos digitales es una tarea ardua y compleja. Conlleva un coste computacional que compromete en cierta medida el procesamiento en tiempo real. Es por esto que en este proyecto se ha usado un sistema de aproximación lineal de la función de transferencia del compresor. Los parámetros de la linealización son calculados previamente en un entorno de computación y almacenados en unas memorias ROM.

La fórmula de aproximación lineal consiste en dividir el rango de valores del eje de abscisas en 2^{n+1} valores, por lo tanto tendremos 2^n espacios (o bandas de linealización), cada uno contendrá una recta que aproxima a la curva de la función potencial en ese espacio.

La aproximación lineal se lleva a cabo mediante la siguiente expresión matemática:

$$\alpha_k = \frac{y_{k+1} - y_k}{x_{k+1} - x_k},\ \beta_k = \frac{x_{k+1}y_k - x_k y_{k+1}}{x_{k+1} - x_k}$$

$$Y = \alpha_k X + \beta_k$$

Los valores de p serán de la forma $K/2^n$ para $K = 1..,2^{n-1}$, en nuestro caso para n=2 (16 intervalos de linealización por cada curva de un valor de p) vamos a obtener una linealización de 15 gráficas para 15 valores distintos del parámetro p.

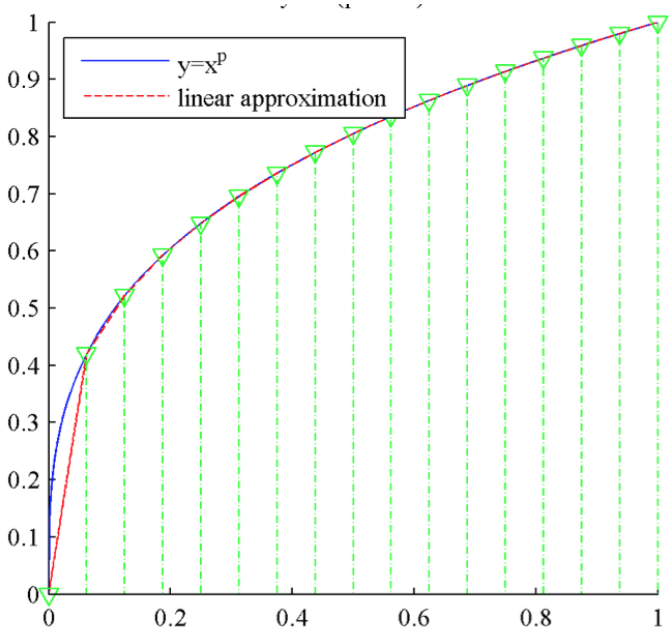


Figura 3.25: Aproximación lineal

En la gráfica (figura 3.26) se muestran las aproximaciones lineales para diferentes curvas potenciales para distintos valores de p (p=1:1:15/16).

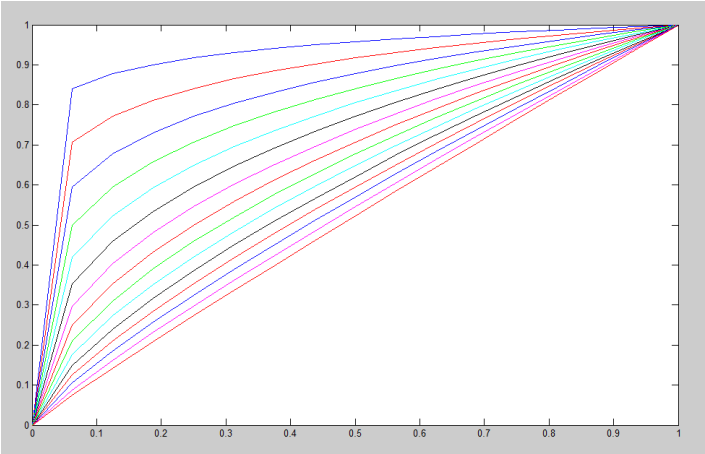


Figura 3.26: Aproximaciones lineales de la función de compresión

Observamos cómo en el primer tramo los coeficientes BETA son siembre nulos ya que las rectas presentan un offset nulo, conforme se van calculando las siguientes rectas de la aproximación lineal estos coeficientes van aumentando.

Con los coeficientes ALFA representan la pendiente de la recta de aproximación y al contrario de lo que ocurre con los coeficientes BETA, estos van decreciendo conforme avanzamos en los intervalos de aproximación ya que se observa una asíntota en y=1.

A la hora de implementar este bloque se contemplan dos estrategias.

-ESTRATEGIA 1: Una estrategia con detección dinámica de las cotas (superior e inferior) de la señal de entrada. Mediante esta estrategia se mantiene actualizado el máximo global del conjunto de máximos que se dan a la entrada del compresor de cada canal. Esto se lleva a cabo mediante un módulo que funciona paralelamente a los veintidós canales. La desventaja es que requiere de un mayor coste en hardware a la hora de implementarla en la FPGA y mayor coste computacional, lo que hace que las simulaciones sean notablemente más lentas. Además, esta estrategia introduce un retardo inicial adicional de aproximadamente 1.2 milisegundos, provocado por los divisores CORDIC, al que hay que sumar el tiempo de aparición del máximo global de la señal de entrada. Este retardo sólo se da al inicio y, debido a que el sistema de detección de máximo alcanza un estado estacionario al cabo de un tiempo, no influye en la latencia del sistema.

-ESTRATEGIA 2: Establecer como parámetros fijos los máximos y mínimos en cada canal. Esta estrategia requiere una fase previa de cálculo de los máximos y mínimos que se dan a la entrada de los compresores en cada canal. Una vez calculados, serán considerados parámetros fijos del sistema. Se consigue evitar el uso de divisores CORDIC debido a que, al ser las amplitudes menores a la unidad, podemos calcular previamente la inversa del máximo y sustituir la operación de división por un multiplicador. Tiene como ventaja la reducción del coste hardware y computacional. La desventaja es claramente un diseño más rígido y menos adaptativo.

En las dos estrategias el diseño en System Generator [10] está compuesto por un bloque de cálculo de dirección de acceso, dos memorias RAM que contienen los coeficientes de la aproximación y un bloque de multiplicación y adición que calcula el resultado de la linealización para cada valor de entrada.

El esquema a implementar se muestra en la figura 3.27

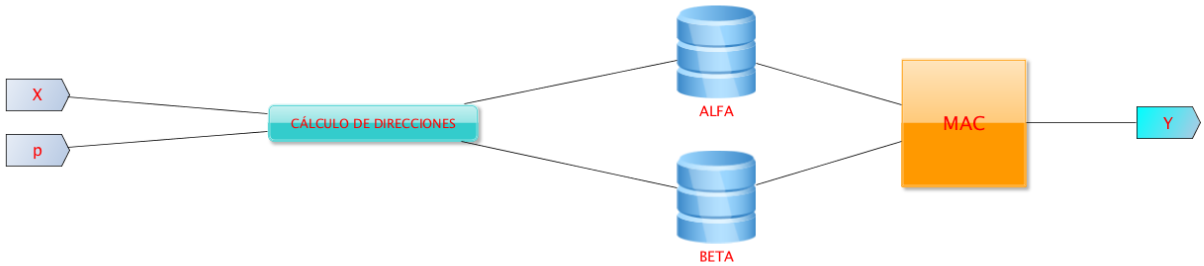


Figura 3.27: Esquema del sistema compresor

3.3.1. Implementación mediante la ESTRATEGIA 1

La función de compresión realizada admite como entrada un valor normalizado de la función entre cero y uno. Para realizar dicha normalización incluiremos en el diseño un bloque que calculará el máximo global de las señales que llegan al compresor de cada canal. Con esto realizaremos un mapeo en iguales condiciones para cada señal de cada canal (normalizada entre 0 y 1). También se incluye en cada compresor un bloque que calcula el mínimo de cada señal de entrada a fin de realizar un offset (de pequeño valor) y eliminar los valores negativos de la señal.

El offset tiene como objetivo evitar los valores negativos, los cuales no tienen sentido porque implicarían una absorción de energía del sistema emisor (no es coherente en términos biológicos ya que el canal sináptico es unidireccional). El bloque de cálculo del mínimo es muy sencillo, la figura 3.28 muestra su implementación.

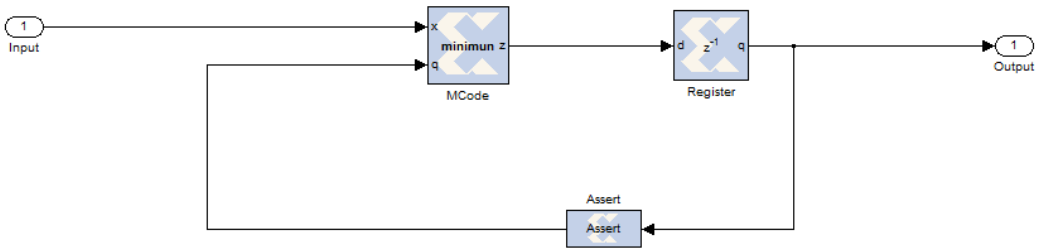


Figura 3.28: Implementación de la estrategia 1

La imagen de la figura 3.29 muestra el bloque que calcula el máximo global de las señales de entrada de los compresores.

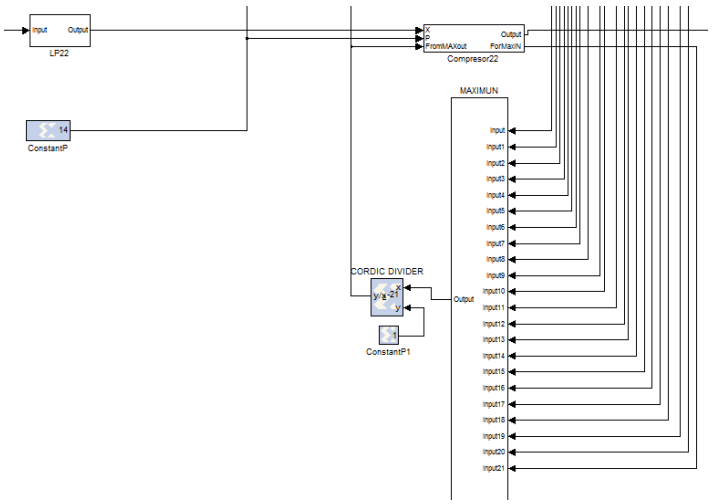


Figura 3.29: Máximo global

Cada entrada del bloque MAXIMO GLOBAL se conecta a un conjunto de bloques MCode que realiza una comparación entre dos entradas. La salida de este bloque la proporciona un subsistema MAXIMUM que es análogo al bloque MÍNIMUM visto anterior mente pero con la función MCode redefinida.

En esencia el bloque de MÁXIMO GLOBAL es un bloque combinacional seguido de un registro con una salida retrolimentada. Esto permite que, una vez detectado el máximo global de los veintidós canales, el bloque proporcione una salida estable y el subsistema alcance un estado estacionario.

En un principio podría pensarse en incluir un divisor por cada módulo Compresor para normalizar la entrada. Esto daría lugar a veintidós divisores, los cuales introducirían en cada canal un retardo aproximado de 1.2 milisegundos.

Sin embargo, debido a que la entrada al sistema esta acotada entre [-1,1], el máximo siempre será un valor menor a la unidad. Gracias a esto podemos sustituir los divisores de cada compresor por multiplicadores. Para ello calculamos la inversa del máximo global. Esto se realiza usando un bloque CORDIC DIVIDER (COordinate Rotation DIvider Computer, ANEXO 1) a continuación del módulo de máximo global. Este bloque permite realizar divisiones usando una estructura de cálculo en paralelo.

La figura 3.30 muestra la configuración interna del bloque Máximo Global.

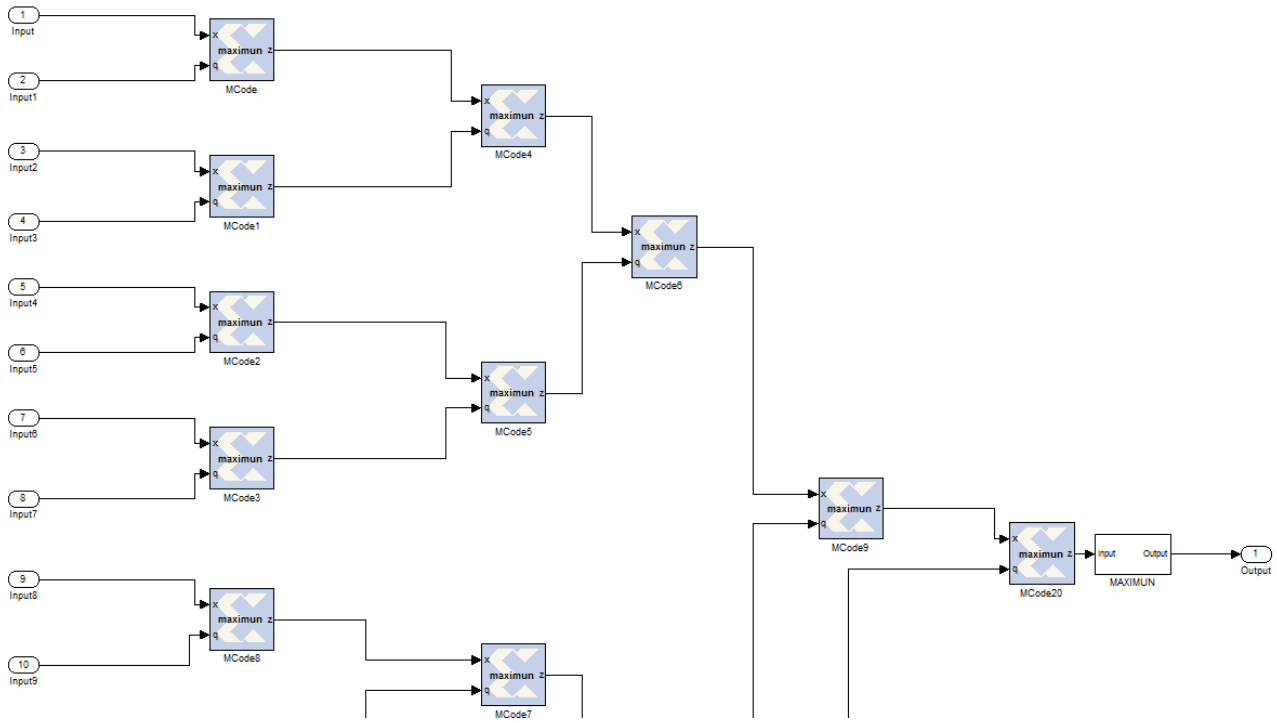


Figura 3.30: Implementación máximo global

El bloque que calcula el Máximo Global tarda un tiempo en alcanzar un estado estacionario. En la figura 3.31 se puede observar la salida del bloque y cómo éste detecta al máximo absoluto del conjunto de señales a su entrada.

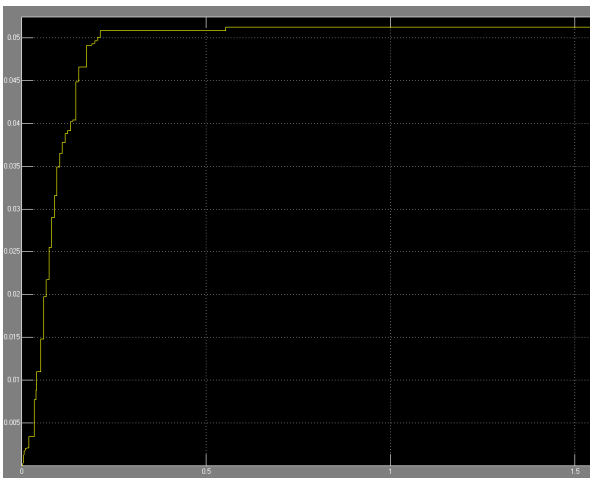


Figura 3.31: Detección del máximo absoluto

Si observamos los bloques Compresor, distinguimos dos puertos que interactúan con el bloque de Máximo Global. Estos son FromMAXout (puerto de entrada) y ForMAXIN (puerto de salida).

La figura 3.32 muestra la implementación interna del bloque Compresor mediante la ESTRATEGIA 1

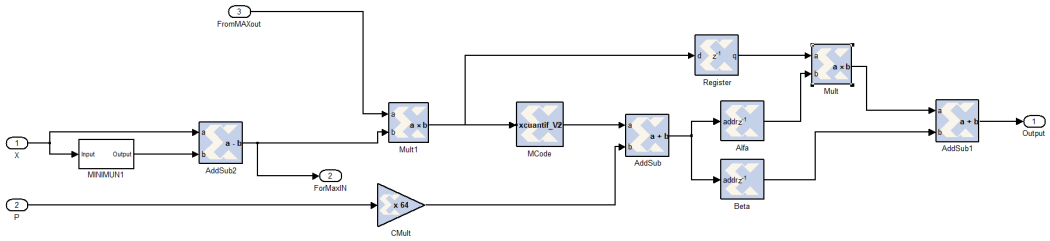


Figura 3.32: Bloque compresor mediante la estrategia 1

El puerto ForMAXin extrae la señal del compresor, una vez se ha realizado el offset, hasta el módulo de máximo global. El puerto FromMAXout importa hacia el bloque compresor el máximo global calculado por dicho módulo. Con este máximo se realiza una normalización de la señal del compresor.

El bloque MCode permite implementar estructuras de selección (en nuestro caso if y elseif) fácilmente. Desarrolla la función de un cuantificador, el cual discriminará en que banda de linealización se encuentra la señal de entrada una vez normalizada (ANEXO 2.1).

La salida va a ser un número entero comprendido entre 0 y $2^n - 1$ y que servirá para calcular la dirección de acceso a memoria en la cual se encuentran almacenados los coeficientes alfa y beta (Memoria ROM, ANEXO 1) de la aproximación lineal.

Los valores del parámetro de compresión p se encuentran entre 1/16 y 15/16. El sistema de compresión recibe como entrada un valor que será el numerador menos uno (n-1), por lo tanto serán valores de 0 a 14.

El cálculo de la dirección de memoria se lleva a cabo de la siguiente forma:

$addr = p' \cdot 2^n + z$

Donde z es un número natural en el intervalo $[0, 2^n - 1]$, el valor cuantificado del valor normalizado de la señal de entrada x.

$p' = p \cdot 2^n - 1$. Es el valor del numerador del factor de compresión p, inicializándolo en cero.

Las dos memorias usadas se cargan con los vectores de coeficientes (alfa y beta) como valores iniciales y una profundidad establecida por la expresión de MATLAB length(alfa) y length(beta) respectivamente. Cabe destacar que cuando se desea acceder a la primera posición de memoria, la dirección de acceso será 0 en lugar del 1 usado en los vectores de MATLAB.
La figura 3.33 muestra la señal de entrada al bloque compresor en el dominio temporal.

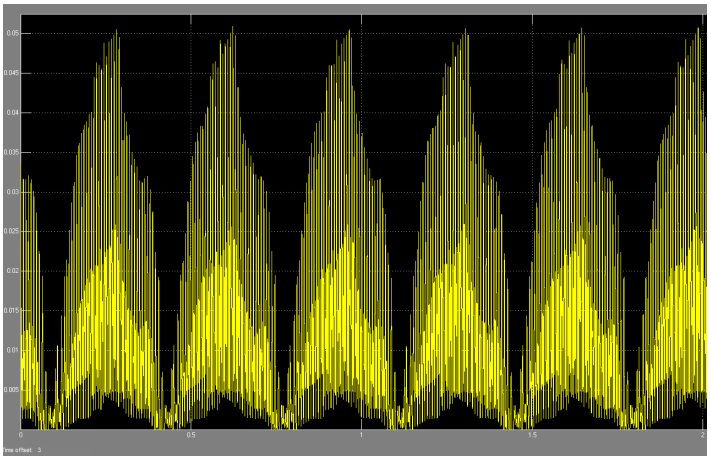


Figura 3.33: Señal de entrada al bloque compresor

En la figura 3.34 muestra la señal de salida del bloque compresor

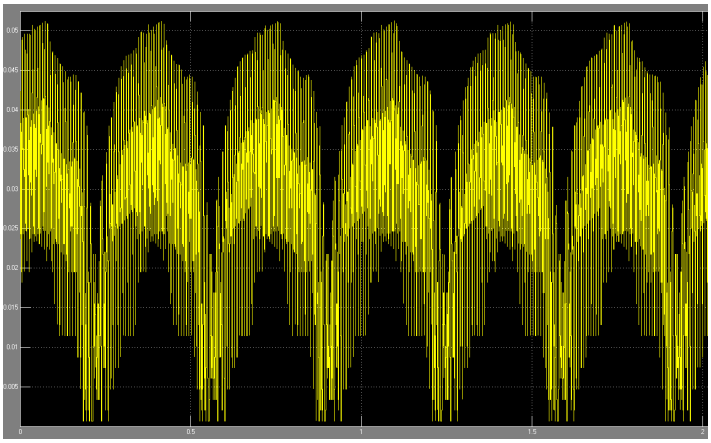


Figura 3.34: Señal de salida del bloque compresor

La señal de salida del compresor presenta una disminución de densidad de puntos para valores bajos del eje de ordenadas. Esto es coherente con la forma de la función de transferencia del compresor.

Cuanto menor sea el factor de compresión, habrá una menor densidad de valores pequeños en la señal. Esto es debido a que la función de transferencia se aproxima a una función potencial con comportamiento asintótico en y=1. Si, el factor de compresión es elevado, es decir próximo a 1, la función de transferencia se aproximará a una recta que pasa por el origen de coordenadas y el punto (1,1).

Cuanto más pequeño es el factor de compresión menor calidad tiene la señal de voz. Esto es debido a que se produce un énfasis en las altas frecuencias, las cuales tienen amplitudes más pequeñas y son trasladadas por el compresor a valores más elevados. La mayor parte de la información de voz se encuentra en frecuencias en torno a los 4 kHz por lo tanto, enfatizar frecuencias superiores a un nivel de amplitud superior va en detrimento de la calidad de la señal.

3.3.2. Implementación mediante la ESTRATEGIA 2

Para calcular previamente los máximos y mínimos que se dan a la entrada del compresor de cada canal, debemos simular antes un esquemático que exporta estos datos al workspace de MATLAB.

Dicho esquemático es una versión del archivo original donde se han sustituido los compresores por bloques CálculoMAXMIN. Estos bloques contienen en su interior los módulos que calculan el máximo y mínimo.

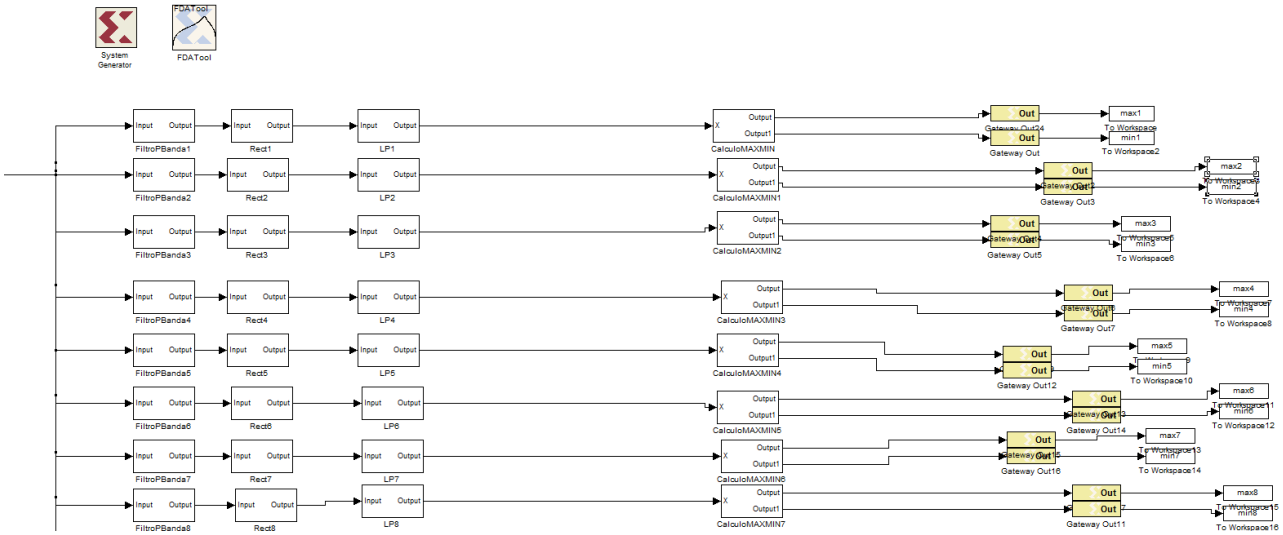


Figura 3.35: Sustitución de los compresores por bloques CalculoMAXMIN

Una vez simulado el esquemático de la figura anterior se procede a ejecutar el modelado del DSP con los compresores modificados.

La figura 3.36 muestra el nuevo diseño (más simple) del sistema compresor.

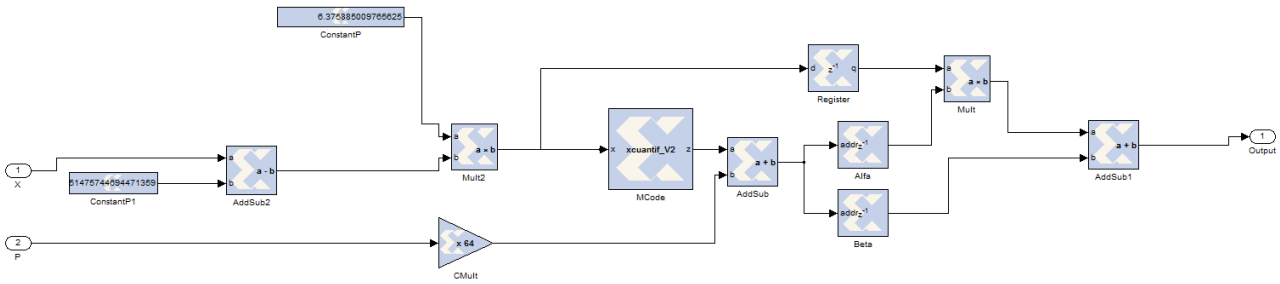


Figura 3.36: Diseño simple del sistema compresor

El uso de constantes y la sustitución del divisor CORDIC por el multiplicador simplifican el diseño y otorgan las ventajas ya comentadas respecto al retardo introducido y a la hora de simular. Exceptuando esta fase inicial del compresor, el resto de diseño y método de funcionamiento es igual al de la primera estrategia.

Capítulo 4

Pruebas y testeo del DSP

INTRODUCCIÓN

Diferenciaremos dos fases en la etapa de prueba y testeo. En primer lugar se realizarán las pruebas del diseño apoyándonos en la simulación por ordenador y aprovechando la integración entre el entorno en aritmética flotante (Simulink) y el entorno en punto fijo (System Generator). Una vez realizadas las pruebas mediante simulación por ordenador se procederá a exportar el diseño a una placa FPGA para estudiar los resultados obtenidos, realizar comparativas y obtener conclusiones finales.

Se usarán el análisis de formantes y el error cometido en la salida de la FPGA frente a los valores de la simulación como parámetros cualitativo de la señal de salida.

El análisis de formantes sirve, entre otros objetivos, para sistemas de reconocimiento de voz. Un sistema de reconocimiento de voz es una síntesis mediante redes neuronales del proceso llevado a cabo por el cerebro. El cerebro humano realiza de forma automática el análisis de formantes, lo que permite una comunicación verbal fluida.

Las formantes se determinan por el filtrado del tracto vocal, que realiza la función de resonador. Estas indican la forma en la que los máximos de energía sonora se distribuyen en frecuencia, son los picos de mayor concentración energética en el espectrograma. Cada vocal y sonido sonoro (un sonido del habla que se produce sin turbulencia en el flujo de aire al atravesar el tracto bucal) pueden ser distinguidos por sus formantes.

En el Español, las cinco vocales pueden ser identificadas con sólo dos formantes. El primer formante (F1, de menor frecuencia) define las vibraciones verticales en la cavidad bucal, el segundo formante (F2) define las vibraciones horizontales en la cavidad y permite identificar si la vocal es anterior, intermedia o posterior.

Dependiendo del tipo de español, existen distintas formantes para la misma vocal. Un sistema de decisión basado en redes neuronales permitirá a un sistema de identificador de voz decidir si un sonido corresponde a una u otra vocal.

La figura 4.1 muestra las formantes F1 y F2 de las cinco vocales existentes en el español.

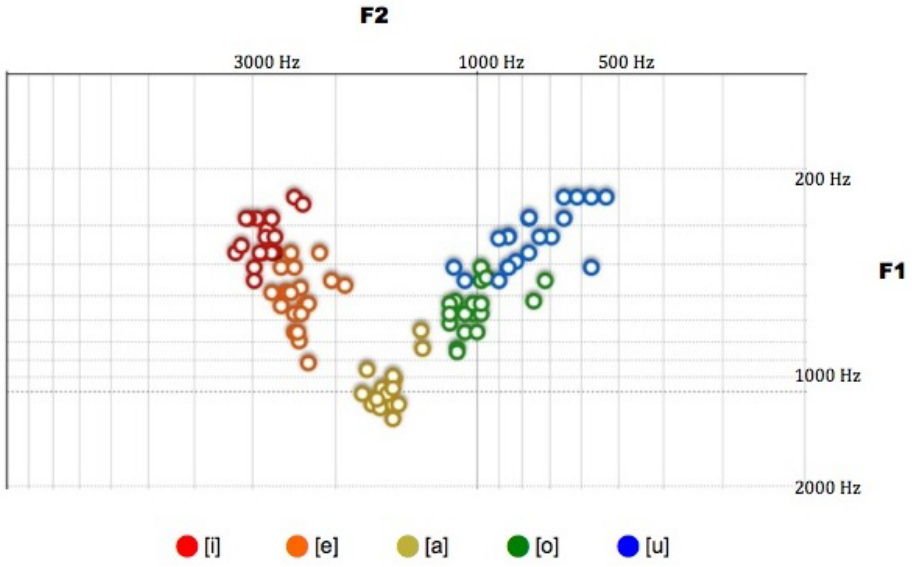


Figura 4.1: Formantes de las vocales

4.1. FASE I: PRUEBAS Y TESTEO MEDIANTE SIMULACIÓN

Los diseños de esta fase se realizarán con bloques Simulink, se trata de una simulación software. En esta fase el diseño de test no va a ser implementado en hardware. El objetivo es la observación y comprobación del funcionamiento del diseño del DSP antes de ser exportado a la FPGA mediante la reconstrucción de la señal de salida.

4.1.1. Recuperación de la señal procesada a la salida del sistema mediante una reconstrucción inversa

Debido a que los filtrados realizados por el DSP no son ideales, siguen existiendo unas componentes frecuenciales remanentes en la banda eliminada. La existencia de estas componentes permite reconstruir la señal revirtiendo el proceso realizado por el DSP.

Una prueba inicial para el sistema consiste en introducir un archivo .wav y observar su salida. El formato .wav fue desarrollado por Microsoft e IBM y consiste en un formato digital de codificación de sonido sin compresión. En internet es poco usado y se usan algoritmos de codificación sin pérdidas como Apple Lossless, que son más eficaces. A pesar del progresivo

desuso de este formato de archivos, debido a características que hacen sencilla su exportación a módulos de System Generator, se usará para el testeo del sistema implementado.

El esquema a seguir para reconstruir el fichero de voz una vez ha sido procesado por el sistema implementado se muestra en la figura 4.2.

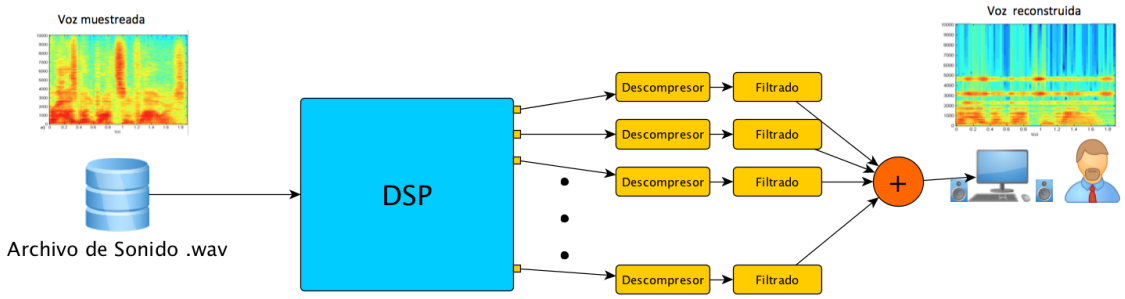


Figura 4.2: Reconstrucción del fichero voz por el método inverso

El archivo .wav que se usa en la simulación es introducido al sistema mediante un bloque ”From multimedia File” (ANEXO 1). El archivo usado presenta una tasa de muestreo de 20000 Hz y el DSP funciona a 22000 Hz por lo tanto existe un sobremuestreo y no se perderá información.

Para regenerar la señal se implementan un descompresor y un banco de filtros, que modelan el proceso inverso llevado a cabo por el DSP, exceptuando la rectificación. En la rectificación se eliminan muestras negativas las cuales son imposibles de recuperar.

El sistema descompresor sirve para devolver la señal a su rango de valores naturales. Para ello se realiza una cuantificación no lineal mediante el código MATLAB embebido en el bloque MATLAB Function (ANEXO 2.5) y se revierte la linealización hecha por el Compresor del DSP.

El filtrado posterior está formado por un paso-alto en serie con un paso-banda centrado en la frecuencia central de cada canal, el objetivo es revertir el filtrado paso-bajo a 200Hz del sistema de detección de envolvente.

La gráfica de la figura 4.3 muestra un zoom del diseño de bloques Simulink del sistema reconstructor a la salida del DSP.

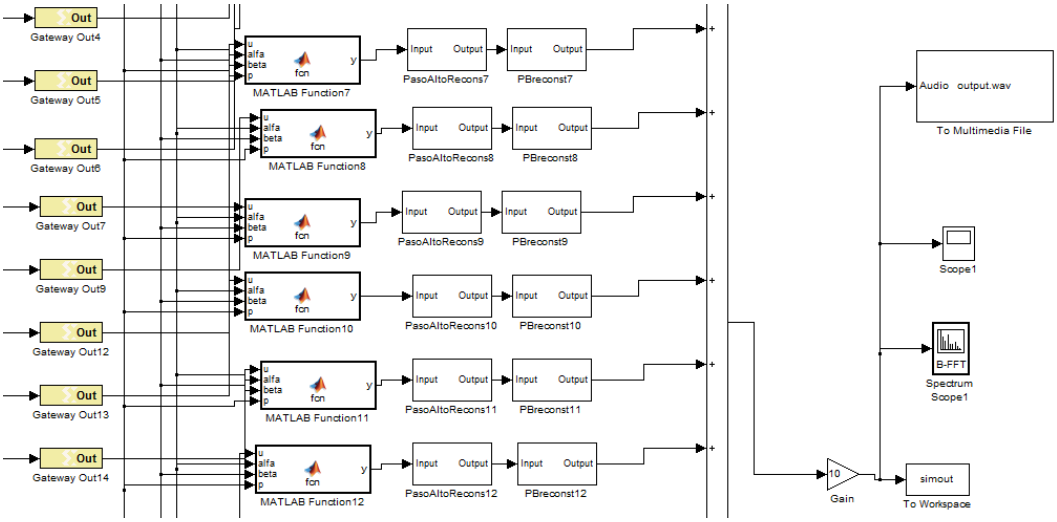


Figura 4.3: Zoom del diseño de bloques Simulink

La figura 4.4 muestra el espectrograma de la señal de voz a la entrada del DSP y la reconstrucción por el método inverso:

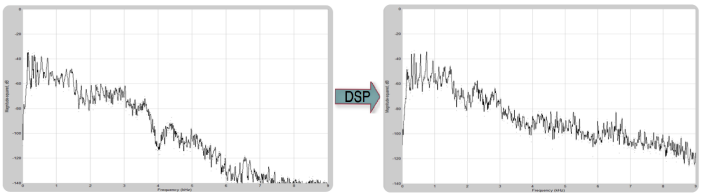


Figura 4.4: Reconstrucción por el método inverso

Los bloques To Workspace almacenan los valores de la señal en simout.signals.values. Ejecutando el comando sound(simout.signals.values, 22000) podemos escuchar la señal de voz reconstruida.

En la imagen (figura 4.5) se muestra la señal de voz en el dominio temporal antes de ser procesada y una vez procesada y reconstruida mediante un proceso inverso al que realiza el DSP.

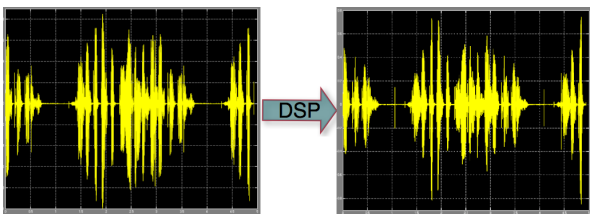


Figura 4.5: Señal de voz en el dominio temporal, original y reconstruida

Formantes de la señal reconstruida mediante el proceso inverso

Las formantes de una señal de voz permiten determinar zonas de concentración energética en determinadas frecuencias.

Un programa que nos permite analizar ficheros de voz es Praat. Se trata de una herramienta de libre distribución para el análisis fonético del habla desarrollada por Paul Boersma y David Weenink en el Instituto de Ciencias Fonéticas de la Universidad de Ámsterdam.

Durante la simulación, incluiremos un bloque Simulink denominado "To Multimedia File" (Anexo 1) . Este bloque se incluye a la salida del sistema reconstructor, nos permite almacenar en el ordenador la señal reconstruida del DSP en formato .wav.

La figura 4.6 muestra el espectro y las formantes del archivo original de voz.

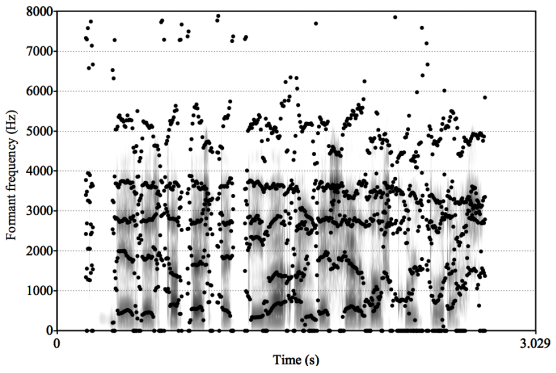


Figura 4.6: Espectro y las formantes del archivo original de voz

En la figura 4.7 se observa el espectro y formantes de la señal reconstruida.

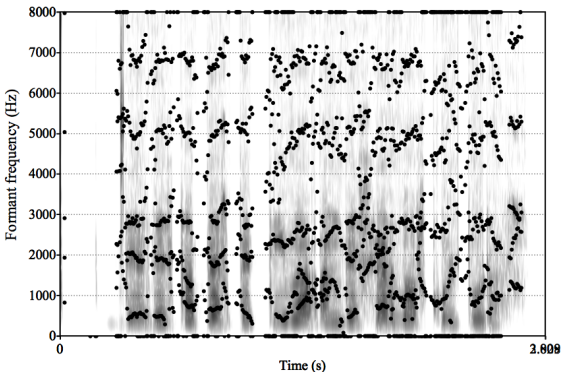


Figura 4.7: Espectro y formantes de la señal de reconstruida

Se puede observar cómo en la reconstrucción aparecen sólo tres formantes inferiores a 4KHz en contraposición a las cuatro formantes de la señal original. Esto da cuenta del

deterioro de la señal, debido a pérdidas de muestras en la rectificación y atenuaciones de los filtros, a pesar del procesado inverso.

4.1.2. Prueba 2: Reconstrucción de la señal de salida mediante modulación

El anterior método solo muestra el proceso inverso al realizado por el DSP. Dicho proceso inverso sólo es posible debido a la no-idealidad de los filtros empleados.

Aunque mediante el anterior método de reconstrucción inversa se consigue recuperar una señal aceptable, no sirve a la hora de reconstruir la señal tal cual la recibe un implantado.

La figura 4.8 muestra el diagrama para la reconstrucción de la señal mediante la modulación con señales sinusoidales.

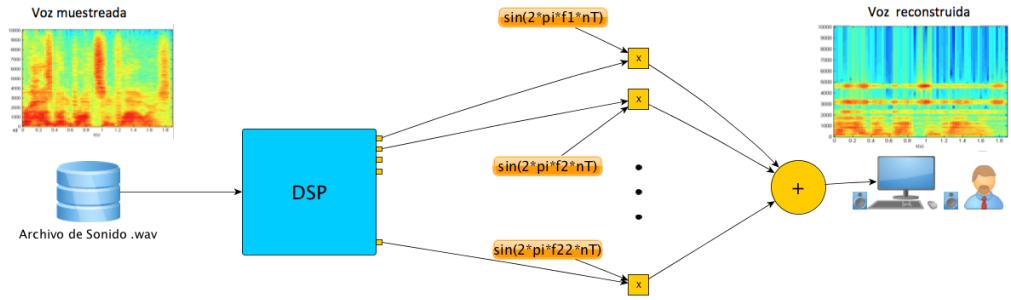


Figura 4.8: Diagrama para la reconstrucción de la señal mediante la modulación con señales sinusoidales

En la "Figura a"podemos observar la señal de salida en el canal veintidós y en la "Figura b"la señal modulada con un seno a dicha frecuencia.

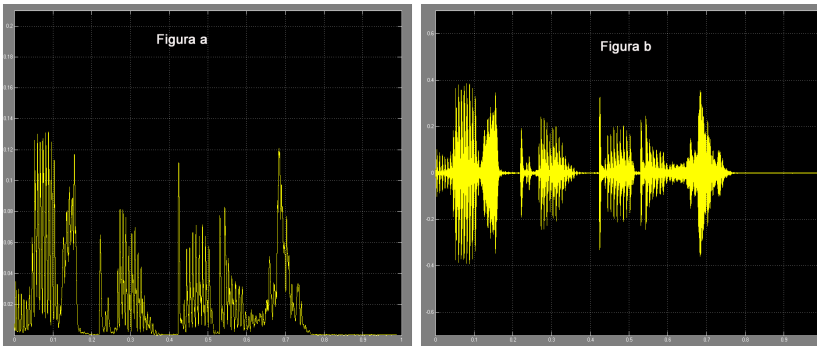


Figura 4.9: Señal de salida en el canal veintidós y modulada con un seno a dicha frecuencia

Observamos en la "Figura a" como los picos donde la señal de salida del canal es mayor, es decir existe mayor energía de señal coinciden en la "Figura b" con una mayor amplitud de señal.

Si mediante esta reconstrucción usamos el comando sound(simout.signals.values, 22000) el sonido que obtenemos es distinto al obtenido mediante la reconstrucción anterior. Se escucha una voz metálica, sin pitch.

La figura 4.10 muestra las formantes sobre el espectrograma de la señal reconstruida mediante modulación con senos. Mediante una comparativa visual con la reconstrucción mediante el método inverso se denota una pérdida de formantes en altas frecuencias así como una dispersión de las mismas.

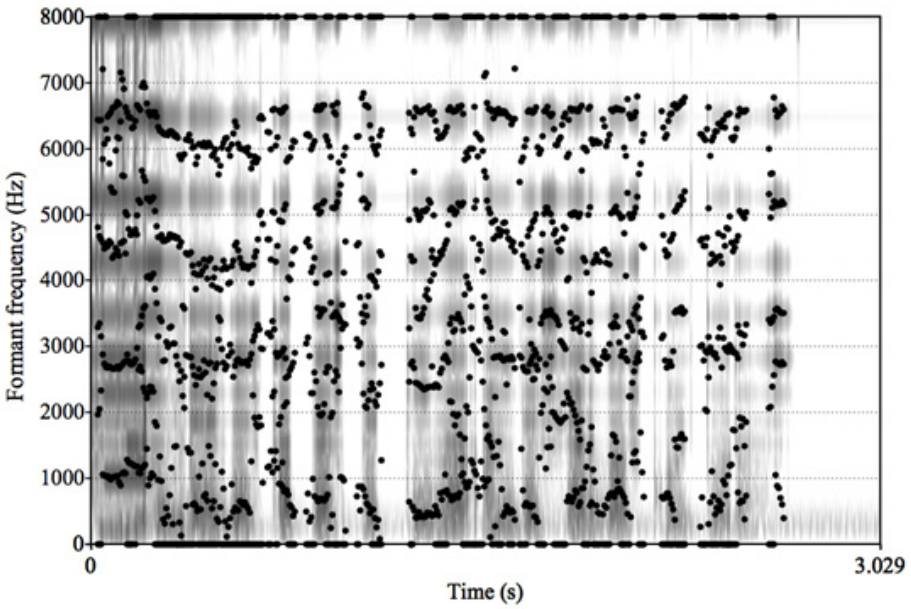


Figura 4.10:

Los implantados no escuchan igual que una persona sana. Esto es debido a la naturaleza discreta del implante. Los filtros pasa-banda tienen como función filtrar la componente frecuencial en una banda concreta para extraer su energía (mediante la detección de envolvente). En este proceso se pierden componentes frecuenciales. El resultado es una señal de baja frecuencia la cual se usará para estimular un área muy limitada de la cóclea.

4.2. FASE II: PRUEBAS Y TESTEO EN LA FPGA

4.2.1. Requerimientos hardware del diseño

El principal problema a la hora de trabajar con System Generator es que el usuario es, en cierta forma, inconsciente de los requerimientos hardware del diseño que implementa. Las ventajas que aporta el tratarse de un sistema de implementación con un elevado nivel de abstracción (superior al de la programación en lenguaje de alto nivel VHDL) hace que la fase de estimación de recursos se postergue a una segunda etapa.

Las FPGAs presentan unos recursos limitados. Esto quiere decir que tendremos que adecuar el diseño a las capacidades del dispositivo al cual queramos exportar el diseño.

System Generator ofrece un bloque llamado Resource Estimator que nos permite realizar una estimación de recursos en una fase previa a la generación de código del diseño. Para comprender los requerimientos que nos proporcionará la suite ISE se deben describir los siguientes elementos:

LUTs: Lookup Tables, son memorias RAM con valores predefinidos que sirven para implementar funciones. La dirección de memoria es el valor de las variables dependientes de la función a implementar y el contenido de la dirección a la que se accede es el valor de la variable independiente.

Existe un compromiso entre área y velocidad a la hora de implementar funciones mediante LUTs. Aveces no es conveniente implementar funciones de muchas variables con una única LUT ya que se reduce el número de slices disponibles en la FPGA.

Aplicando sucesivas veces el teorema de Shannon podemos implementar cualquier función con un número determinado de variables con LUTS de un número inferior de variables.

Funcion de Shannon.

$$F(X_1, X_2, ..., KX_n) = X_1 \cdot F(1, X_2, ..., KX_n) + \overline{X_1} \cdot F(0, X_2; ..., KX_n)$$

La figura 4.11 representa el compromiso entre el Área ocupada en la implementación y el número de variables usadas para cada LUT

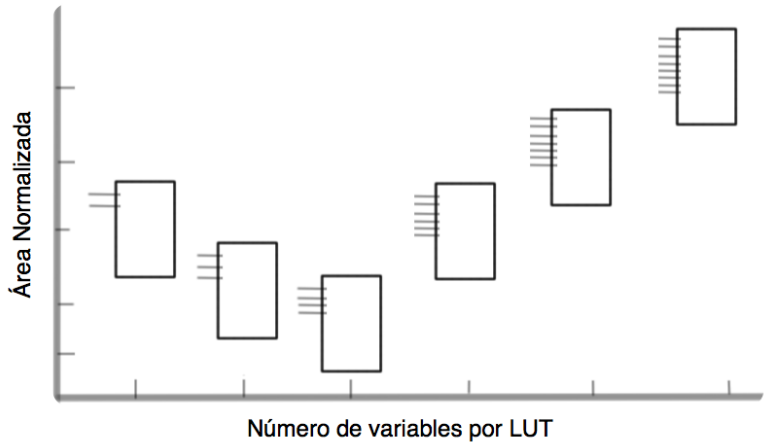


Figura 4.11: Área ocupada en la implementación y número de variables usadas para cada LUT

El uso de LUTs con pocas variables compromete la rutabilidad del diseño, incrementando los retardos de propagación, como puede observarse en la figura 4.12

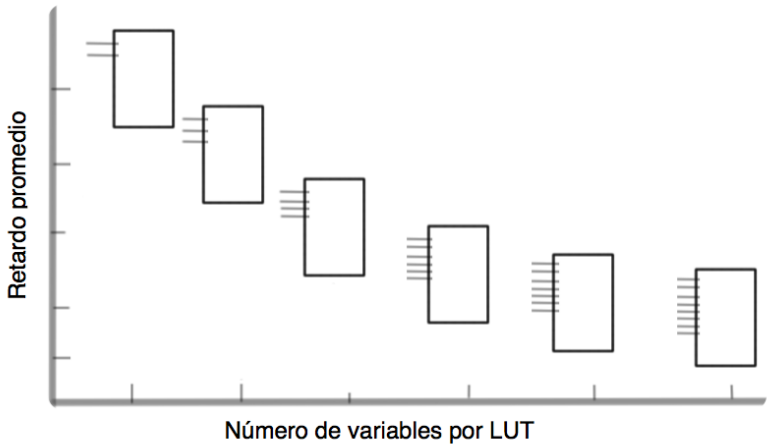


Figura 4.12: Retardos de propagación

Slices: son los bloques lógicos programables elementales en las familias Virtex-4 y anteriores. Dichos bloques constan de:

-2 LUT (look up table) de cuatro entradas, capaces de implementar funciones booleanas de hasta cuatro variables.

-Dos multiplexores, MUXFX y MUXF5. MUXF5 sirve para combinar las salidas de las dos LUTs. Por lo tanto permite implementar cualquier circuito combinacional de 5 entradas. MUXFX sirve para combinar las salidas de un MUXFX y MUXF5 de otro bloque lógico programable.

-Un bloque aritmético lógico que consta de dos sumadores de un bit (conectados por acarreo) y dos puertas AND.

-Dos registros de 1 bit, que pueden funcionar como latches o flip-flops. Las entradas de estos registros están seleccionadas por los multiplexores XMUX e YMUX, que no son controlables por el usuario.

En los slices se encuentra la mayor parte de la funcionalidad de la FPGA. Se suelen agrupar de 2 en 2 o de 4 en 4 formando bloques lógicos configurables (CLBs).

La figura 4.13 muestra un Slice de una FPGA Virtex-4:

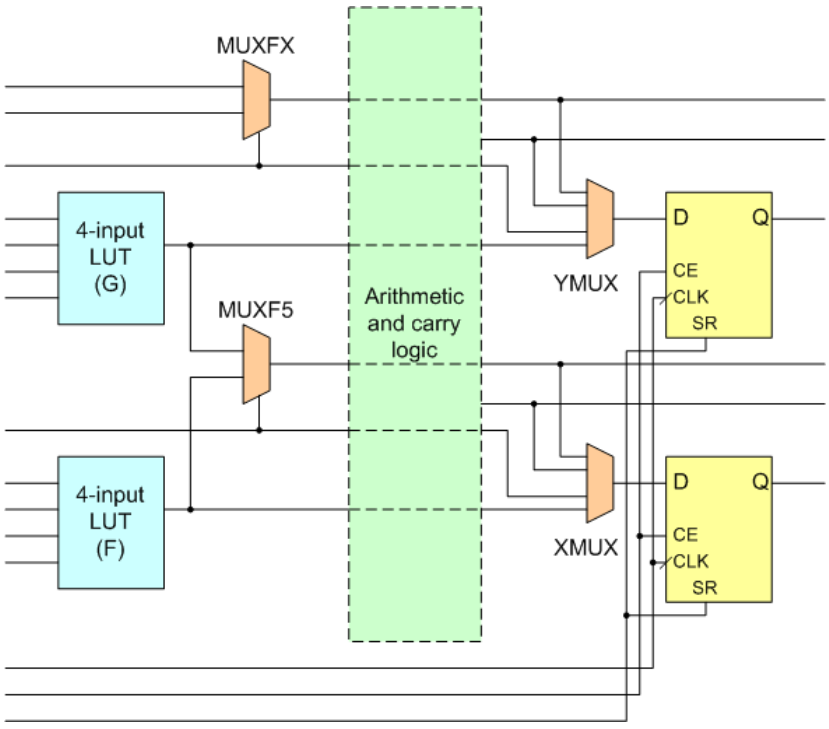


Figura 4.13: Slice de una FPGA Virtex-4

FFs: indica el número de flip-flops que hay en nuestro modelo hardware, estos componentes almacenan un bit durante un ciclo de reloj y pueden ser activados por flanco de subida o de bajada.

BRAMs: La opción más rápida de implementar memorias es usar esta memoria local. El acceso a la misma es de un ciclo de reloj. El tamaño de esta memoria local depende de la versión de FPGA usada.

IOBs: Los bloques de entrada/salida de la FPGA son análogos a las macroceldas de otros dispositivos electrónicos pero con más controles lógicos. Disponen de configuraciones de entrada y salida combinacionales o registradas, alta impedancia, elementos de retardo y con-

troles analógicos.
DSP48: Este circuito integrado, incluido en las últimas familias de Xilinx, puede implementar funciones matemáticas básicas de manera sencilla y logrando alta velocidad. Puede implementar sumadores/restadores, acumuladores, varios tipos de multiplicadores, multiplicadores/acumuladores (MAC), multiplexores, desplazadores, contadores, divisores y raíces cuadradas. Así, la asociación de varios de estos circuitos se presenta como un recurso muy potente para la implementación de funciones avanzadas de manera eficiente.

El bloque Resource Estimator está relacionado con el bloque System Generator en tanto que su estimación se basa es un análisis Post-Map, el cual depende del modelo de FPGA seleccionada.

En la figura 4.14 observamos la estimación de recursos para la Virtex 4 XC4VLX160.

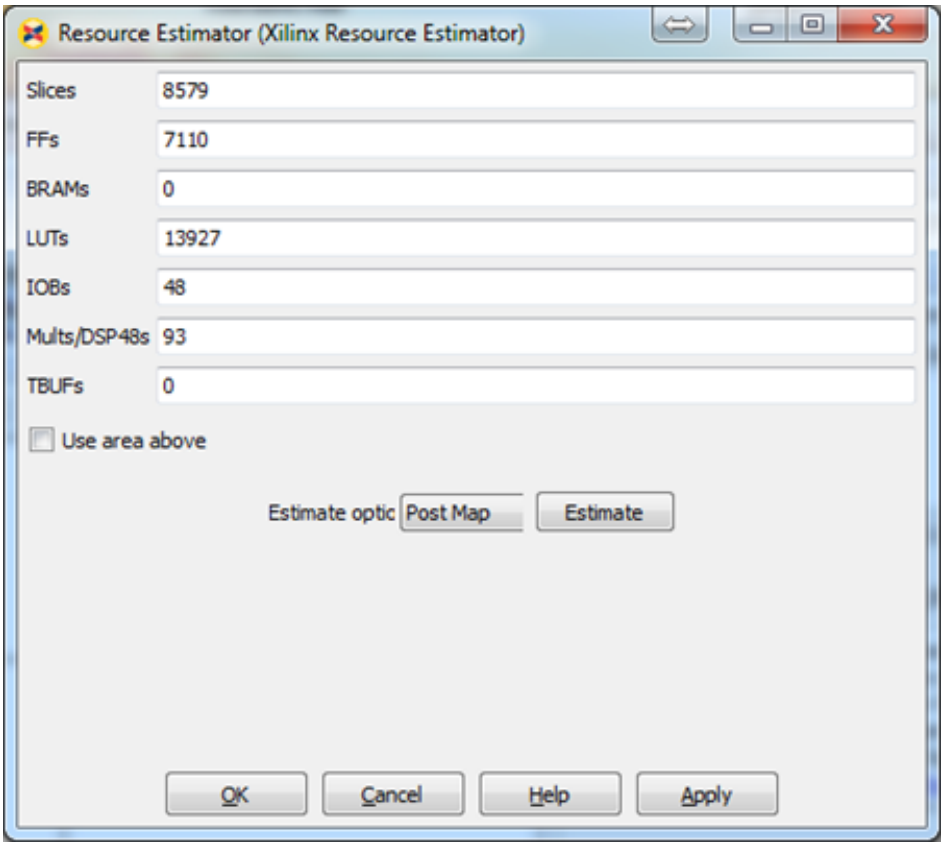


Figura 4.14: Recursos para la Virtex 4

Podemos comparar los datos obtenidos con la tabla de especificaciones de este modelo de Virtex4. Observamos como el número de DSP48 no sobrepasa el número disponible, que es de 96.

Table 1: **Virtex-4 FPGA Family Members**

Device	Configurable Logic Blocks (CLBs)(1)				XtremeDSP Slices(2)	Block RAM		DCMs	PMCDs	PowerPC Processor Blocks	Ethernet MACs	RocketIO Transceiver Blocks	Total I/O Banks	Max User I/O
	Array(3) Row x Col	Logic Cells	Slices	Max Distributed RAM (Kb)		18 Kb Blocks	Max Block RAM (Kb)							
XC4VLX15	64 x 24	13,824	6,144	96	32	48	864	4	0	N/A	N/A	N/A	9	320
XC4VLX25	96 x 28	24,192	10,752	168	48	72	1,296	8	4	N/A	N/A	N/A	11	448
XC4VLX40	128 x 36	41,472	18,432	288	64	96	1,728	8	4	N/A	N/A	N/A	13	640
XC4VLX60	128 x 52	59,904	26,624	416	64	160	2,880	8	4	N/A	N/A	N/A	13	640
XC4VLX80	160 x 56	80,640	35,840	560	80	200	3,600	12	8	N/A	N/A	N/A	15	768
XC4VLX100	192 x 64	110,592	49,152	768	96	240	4,320	12	8	N/A	N/A	N/A	17	960
XC4VLX160	192 x 88	152,064	67,584	1056	96	288	5,184	12	8	N/A	N/A	N/A	17	960
XC4VLX200	192 x 116	200,448	89,088	1392	96	336	6,048	12	8	N/A	N/A	N/A	17	960

Figura 4.15: Tabla de especificaciones del modelo Virtex4

Si seleccionamos un modelo de FPGA con características inferiores a la Virtex4 XC4VLX160 no obtenemos una estimación de recursos puesto que el proceso de mapeo (”Map”) indica que se sobrepasan los recursos disponibles. Es en la fase de packaging cuando se lanza el error al no poder implementar el diseño con los recursos disponibles. El error es lógico ya que, si nos fijamos en la hoja de características de la Spartan 3E, podemos observar como sólo dispone de 20 multiplicadores 18x18, una cantidad muy inferior a la requerida.

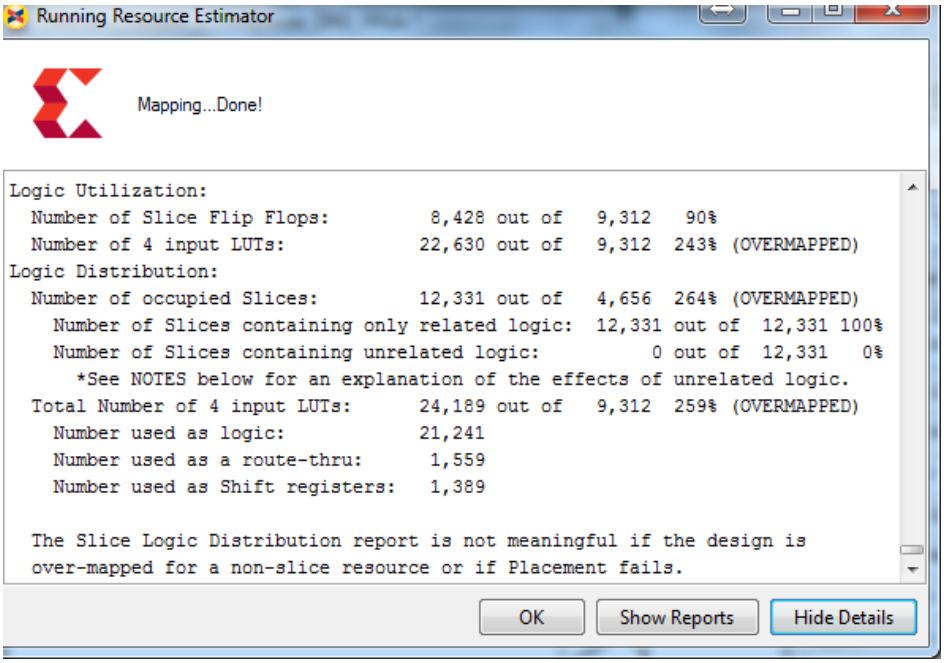


Figura 4.16: Características de Spartan 3E

4.2.2. Generación del código HDL y los ficheros de síntesis e implementación

Esta fase consiste en exportar el diseño a una FPGA para que sea esta la que realice el procesado de la señal. Para ello se generará el código HDL correspondiente al diseño y se usará la suite ISE Project Navigator.

A la hora de generar el código HDL hay que seleccionar el bloque "System Generator"del esquemático. En la figura 4.17 se puede observar el menú de configuración de dicho bloque.

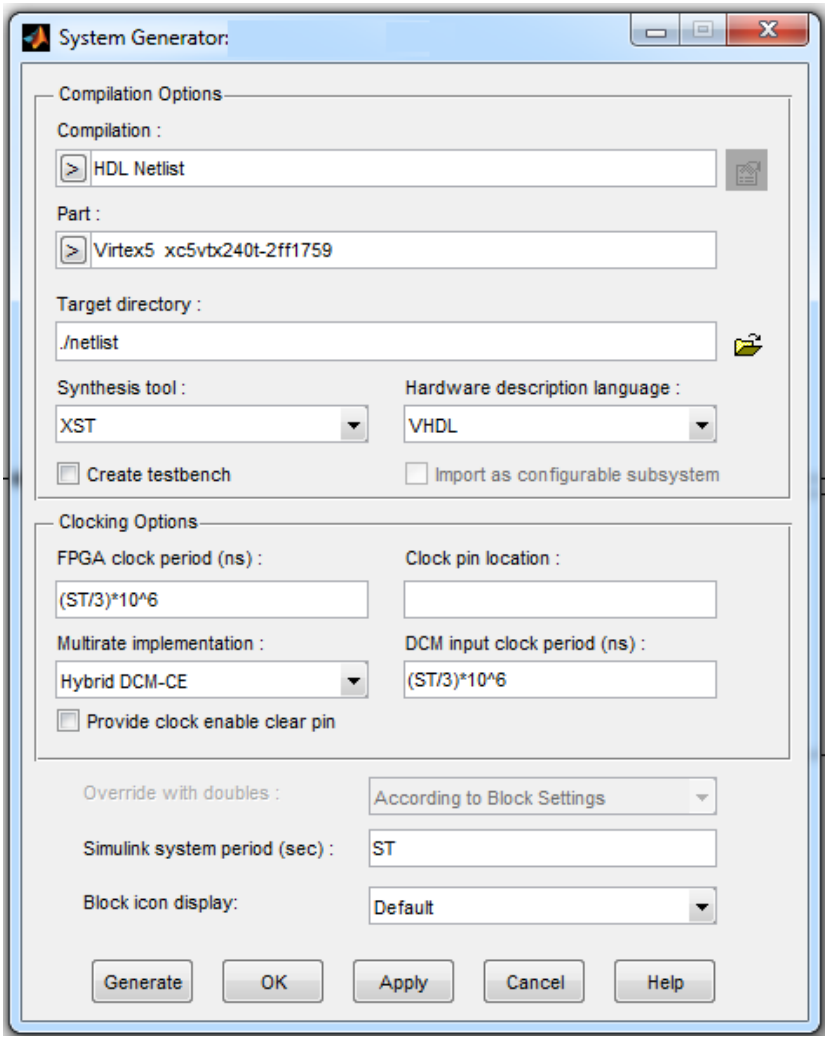


Figura 4.17: Menú de configuración del bloque System Generator

En el campo " Compilation " se selecciona "HDL netlist". HDL es el acrónimo de "Hardware Description Language". HDL consiste en un lenguaje de computación capaz de describir formalmente y de forma precisa el comportamiento y la estructura de circuitos lógicos digitales. HDL es un lenguaje de programación de alto nivel consistente en expresiones, es-

tados del sistema y estructuras de control y que, a diferencia de otros lenguajes, tiene en cuenta la variable del tiempo.

En el campo "Part"seleccionamos el dispositivo en el cual vamos a exportar el diseño. La versión de System Generator usada en este Proyecto Fin de Carrera ofrece la posibilidad de exportar el proyecto para FPGAs hasta los modelos Spartan6 y Virtex6 y sus distintas versiones.

En "Synthesis Tool"seleccionamos XST, que es el acrónimo de Xilinx Synthesis Tool, una herramienta de síntesis incluida en la suite ISE. Esta herramienta genera los archivos de visión RTL (de extensión .NGR) de visor de tecnología y herramientas de implementación (de extensión .NGC) y archivos de información (.LOG). Todos estos archivos son generados a partir de las librerías IP Core (Intellectual Property Core), constraints (archivo de restricciones) y los ficheros VHDL o Verilog.

Una vez pulsemos el botón "Generate", se creará una carpeta en la ubicación seleccionada (por defecto en el fichero raíz y con nombre de carpeta /netlist) que contiene los ficheros necesarios para trabajar con la suite ISE antes de exportar el diseño a la FPGA.

En el campo Çlocking Options"se selecciona el periodo de trabajo de la FPGA, que nunca podrá ser inferior a la inversa de la frecuencia máxima de funcionamiento, que dependerá del modelo y de la configuración de la FPGA. En "Multirate Implementations"podemos seleccionar entre DCM (Digital Clock Manager) o Çlock Enables". Este campo sirve cuando los componentes del diseño funcionan a distintas frecuencias.

4.2.3. Depuración y optimización del diseño

La estrategia de optimización se focaliza en la modificación de estructuras ya diseñadas y en la restricción de los bits necesarios para la cuantificación en cada etapa con el fin de optimizar el uso de recursos disponibles sin ir en detrimento de la calidad ofrecida por el sistema.

-Modificación en los filtros Pasa-Banda y Paso-Bajo:

A pesar de que los recursos consumidos por estos filtros son en un principio elevados, se descarta usar los bloques MAC-FIR de las IPCore proporcionados por System Generator. Esto es debido a que el orden del filtro FIR necesario para conseguir las prestaciones obtenidas con un IIR se eleva a ordenes de magnitud de 10^3 y producen un retardo en el sistema inasumible o, en el mejor de los casos, no deseado.

Para reducir el consumo de recursos de estos filtros restringe el número de bits de las entradas del multiplicador mediante bloques CONVERT y se introduce el retardo de la realimentación en los propios multiplicadores. La estructura final del filtro se muestra en la figura 4.18.

Como puede observarse, en este diseño aparece un retardo en los multiplicadores. El introducir este retardo en los bloques multiplicadores y no mediante un bloque Delay se traduce en que dicho retardo es asumible por la parte lógica (logic) de la implementación y

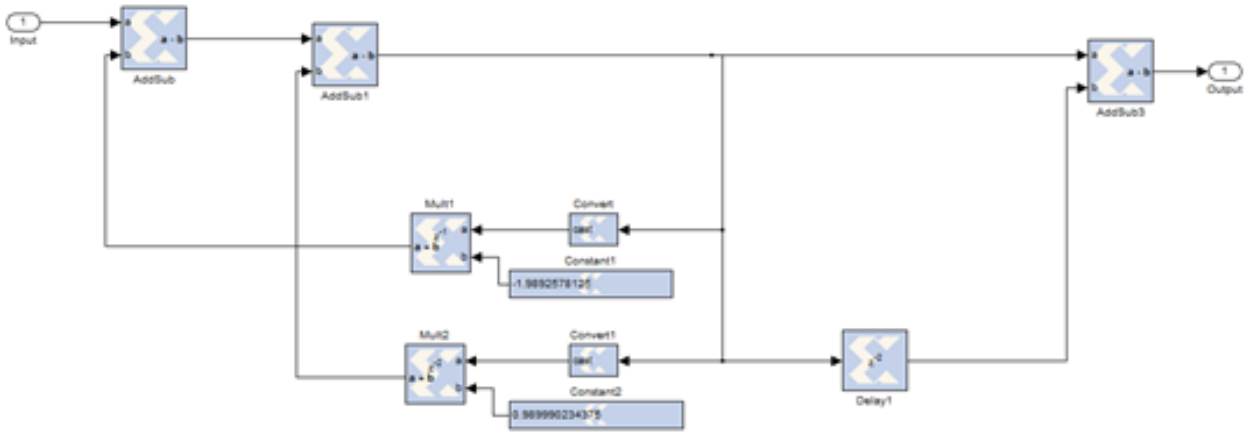


Figura 4.18: Los filtros Pasa-Banda y Paso-Bajo

no por el ruteado (path). Esto produce una relajación en las exigencias de implementación. Dicho retardo aparece indicado con z^{-1} y z^{-2} en los bloques de diseño.

Además se introducen los bloques Cast, con los cuales podemos limitar el número de bits a la entrada de los multiplicadores. Además se redujo la resolución de los coeficientes de los filtros con el fin de que éstos pudiesen ser implementados con los bloques DPS48 disponibles.

Comprobando las resoluciones necesarias para no perder calidad en la reconstrucción de la señal, se establecen 20 bits con el punto decimal en 18 para las entradas ”a” de los multiplicadores y 14 bits con el punto decimal en 12 para los coeficientes.

Para no perder la perspectiva de funcionamiento de los filtros paso banda con la anterior estructura, se comprobó su funcionamiento y respuesta en frecuencia usando el esquema ya utilizado en el apartado 3.1.3, ”FDATool en el diseño de filtros digitales”.

-Modificación en la etapa de compresión:

En el diseño inicial se estableció un compresor por cada canal del sistema. El compresor es el bloque que más recursos hardware consume debido a las dos memorias ROM y los multiplicadores.

El bloque compresor, a diferencia de los filtros, no opera con valores retardados de la señal. Gracias a esto puede usarse un mismo compresor para todo el sistema. La estrategia consiste en serializar los valores de salida de cada canal e introducirlos por un mismo compresor. Esto dará lugar a un flujo de valores comprimidos a la salida de dicho bloque.

Para que la anterior estrategia sea viable es necesario establecer una frecuencia de funcionamiento del sistema de, como mínimo, ST/n. Donde ST es el periodo de muestreo y n el número de canales del sistema.

Además de la multiplexación temporal para compartir el compresor, se prescindirá del multiplicador que normaliza la señal de entrada. Ahora esta señal será mapeada por el bloque MCode entre 0 y un máximo calculado previamente con el módulo Máximo Global para ocho canales.

En las memorias ROM de coeficientes Alfa y Beta se cargarán únicamente los parámetros correspondientes al factor de compresión que se desee usar. En el caso de una cuantificación de 16 bandas pasaremos de una memoria de 16bits x 240 posiciones de memoria a una memoria de 16bits x 16 posiciones de memoria.

La figura 4.19 muestra el resultado del módulo de compresión una vez realizados las citadas modificaciones.

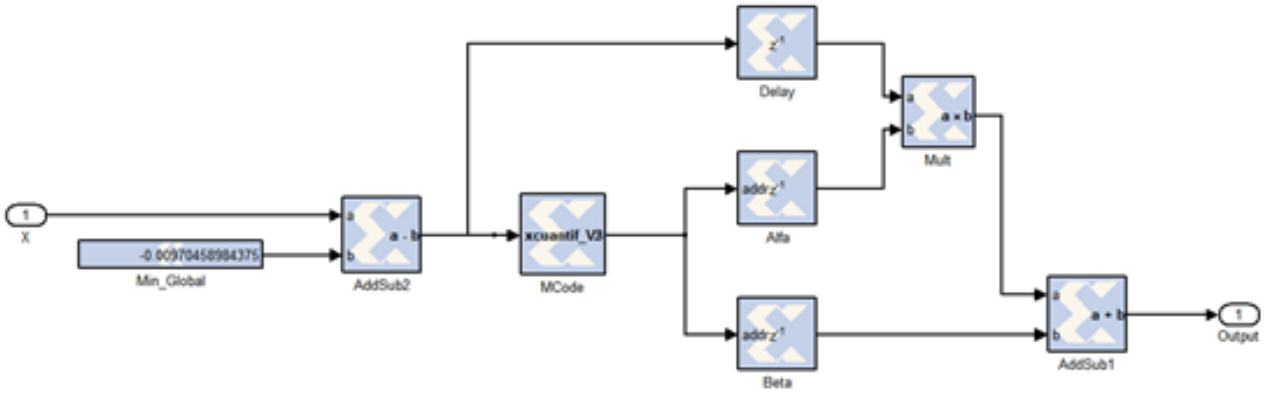


Figura 4.19: Módulo de compresión con las modificaciones

Puede optarse también por una implementación que incluya la detección de rango dinámico (DRD), en lugar de establecer unos parámetros de máximo y mínimo global. En este caso, el mapeo se realizará entre 0 y 1. Será necesario la inclusión del multiplicador para la normalización de la señal de entrada así como dos bloques de cálculo de cotas superior en inferior y un divisor CORDIC. Al multiplexar temporalmente los canales de salida los bloques de detección de cotas son más sencillos, ya que realizan la comparación secuencialmente. La figura 4.20 muestra la implementación del bloque compresor y de la detección dinámica de rango.

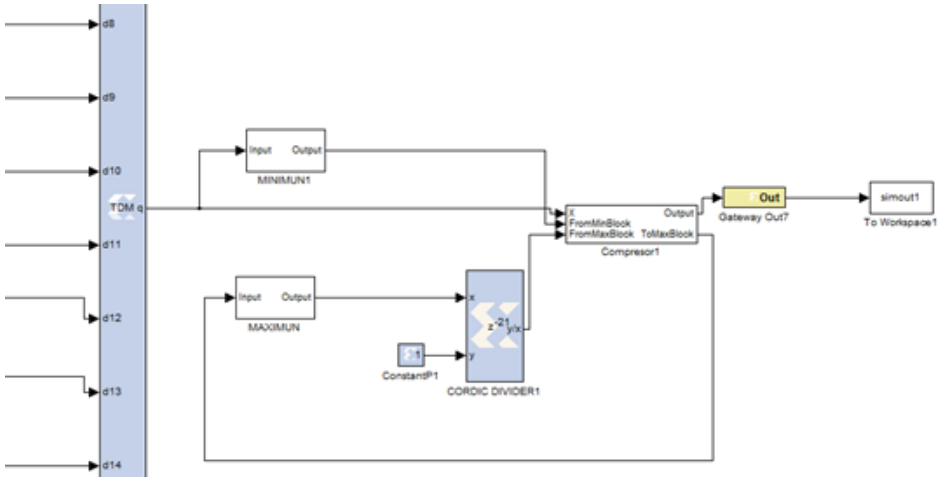


Figura 4.20: Bloque compresor. Detección dinámica de rango

En la figura 4.21 podemos observar la estimación Post-Map de recursos usando la DRD y en la figura 4.22 la estimación Post-Map prescindiendo de la misma.

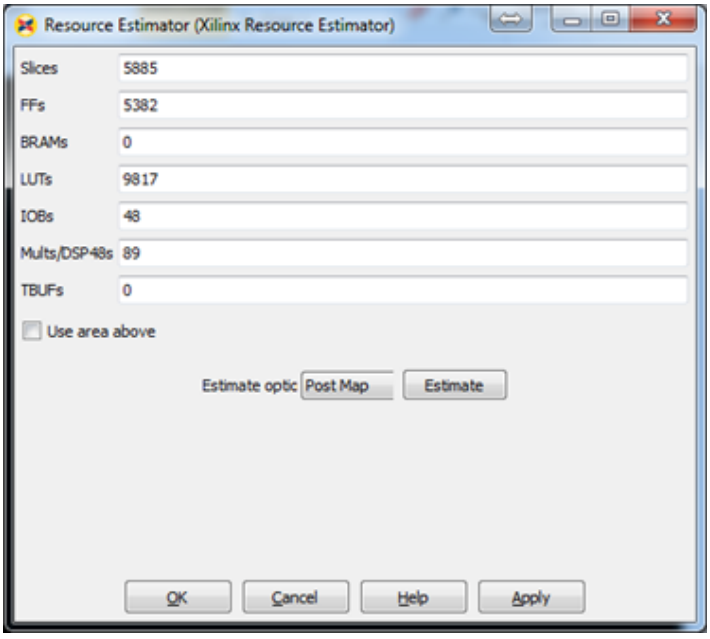


Figura 4.21: Figura a

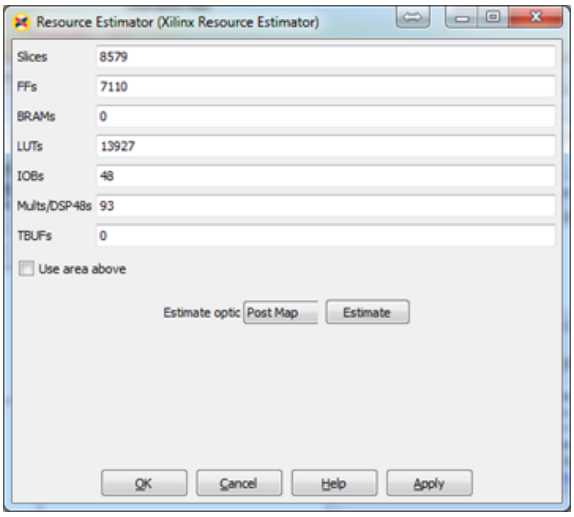


Figura 4.22: Figura b

Se puede optar por un número más elevado de bits para implementar los multiplicadores. En este caso, debido a que el uso de los DSP48 sobrepasa la disponibilidad de la FPGA, parte de los multiplicadores son implementados por la placa mediante otras configuraciones disponibles.

La figura 4.23 muestra los recursos necesarios cuando el número de bits del multiplicando se elevan a veinticuatro. Se observa como, ante la imposibilidad de implementar todos los multiplicadores mediante DSP48, se implementan a costa de otros recursos disponibles como LUTs y SLICES.

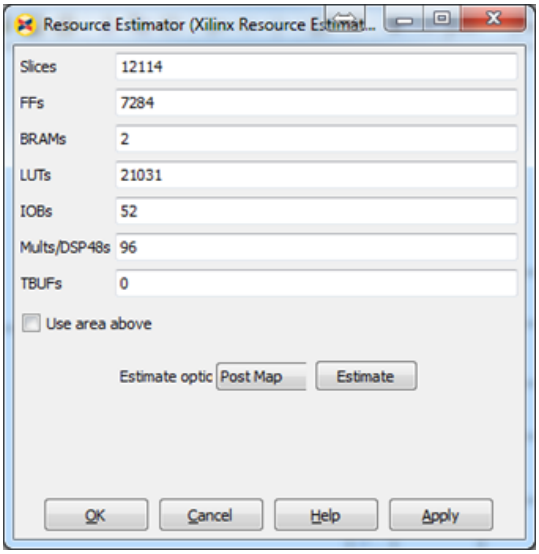


Figura 4.23: Implementación por medio de LUTs y SLICES

La implementación alternativa a los DSP48 obliga a reducir la frecuencia de funcionamiento del diseño, ya que una de las restricciones temporales se incumple. Esto puede ob-

servarse en el apartado ”Time Constrains” de ISE Proyect Navigator, como muestra la figura 4.24.

	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	No	TS_clk_da7d685e = PERIOD TIMEGRP "clk_da7d685e" 50 ns HIGH 50%	SETUP HOLD	-5.344ns 0.440ns	55.344ns	16 0	73386 0
2	Yes	COMP "gateway_out7(31)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	16.559ns	33.441ns	0	0
3	Yes	COMP "gateway_out7(29)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	16.992ns	33.008ns	0	0
4	Yes	COMP "gateway_out7(27)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.199ns	32.801ns	0	0
5	Yes	COMP "gateway_out7(26)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.226ns	32.774ns	0	0
6	Yes	COMP "gateway_out7(28)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.360ns	32.640ns	0	0
7	Yes	COMP "gateway_out7(24)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.512ns	32.488ns	0	0
8	Yes	COMP "gateway_out7(22)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.654ns	32.346ns	0	0
9	Yes	COMP "gateway_out7(20)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.734ns	32.266ns	0	0
10	Yes	COMP "gateway_out7(15)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.938ns	32.062ns	0	0
11	Yes	COMP "gateway_out7(30)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.952ns	32.048ns	0	0
12	Yes	COMP "gateway_out7(23)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.967ns	32.033ns	0	0
13	Yes	COMP "gateway_out7(25)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	17.981ns	32.019ns	0	0
14	Yes	COMP "gateway_out7(13)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.072ns	31.928ns	0	0
15	Yes	COMP "gateway_out7(21)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.125ns	31.875ns	0	0
16	Yes	COMP "gateway_out7(18)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.132ns	31.868ns	0	0
17	Yes	COMP "gateway_out7(12)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.285ns	31.715ns	0	0
18	Yes	COMP "gateway_out7(19)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.325ns	31.675ns	0	0
19	Yes	COMP "gateway_out7(14)" OFFSET = OUT 50 ns AFTER COMP "clk"	MAXDE...	18.336ns	31.664ns	0	0

Figura 4.24: Restricciones temporales (Time Constraints)

En la figura 4.25 se observan los recursos utilizados en la FPGA para distintas configuraciones de bits de cuantificación por canal.De 12 a 20 bits se han usado 92 DSP48 de los 96 disponibles. De 22 bits en adelante se usan los 96 DSP48 disponibles

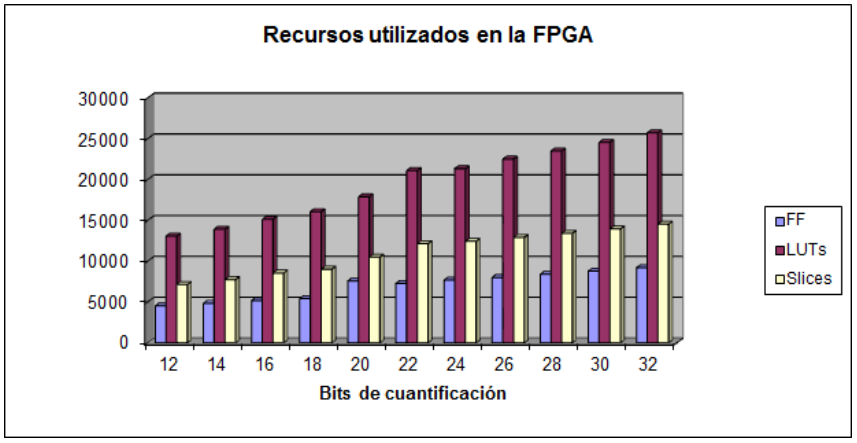


Figura 4.25: Recursos utilizados en la FPGA

4.2.4. Selección del modelo de FPGA e implementación del diseño

El diseño implementado es lo suficientemente exigente en recursos como para descartar el uso de placas FPGA starter como la Spartan3E. Por motivos de disponibilidad se decide

trabajar con el modelo Virtex4 XC4VLX160-10FF1513 de AVNET. Se trata de un Kit de desarrollo de carácter profesional, con un coste muy superior al de la Spartan3E.

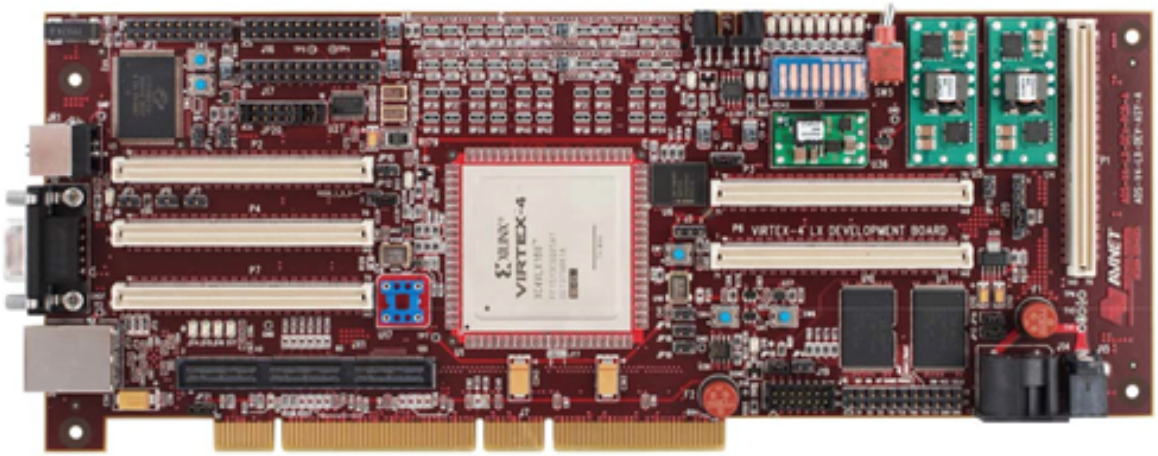


Figura 4.26: Modelo Virtex4 XC4VLX160-10FF1513 de AVNET

Una vez seleccionado este modelo en el bloque System Generator y generados los datos del modelado, realizamos la síntesis con la suite ISE Project Navigator. En la siguiente tabla se muestra una estimación previa de la utilización de recursos del sistema.

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	9157	67584	13%
Number of Slice Flip Flops	7996	135168	5%
Number of 4 input LUTs	16063	135168	11%
Number of bonded IOBs	53	960	5%
Number of GCLKs	1	32	3%
Number of DSP48s	96	96	100%

Figura 4.27: Estimación con la suite ISE Project Navigator

La siguiente fase consiste en la implementación del diseño, dicha fase consta de tres procesos: Translate, Map y PlaceRoute.

-Translate: se genera un archivo que engloba los netlist (descripción de la conectividad del circuito) y archivos NGC generados por el proceso de síntesis y Coregen.

-Map: transforma las puertas genéricas y los biestables en bloques concretos del modelo FPGA usado. Es en esta parte del proceso cuando se conoce realmente la utilización de recursos hardware del diseño.

La figura 4.28 muestra la tabla de la utilización final de recursos que hace el diseño de veintidós para la FPGA Virtex 4 usada.

Device Utilization Summary				
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	7,378	135,168	5%	
Number of 4 input LUTs	20,239	135,168	14%	
Number of occupied Slices	11,681	67,584	17%	
Number of Slices containing only related logic	11,681	11,681	100%	
Number of Slices containing unrelated logic	0	11,681	0%	
Total Number of 4 input LUTs	21,546	135,168	15%	
Number used as logic	18,676			
Number used as a route-thru	1,307			
Number used as Shift registers	1,563			
Number of bonded IOBs	53	960	5%	
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Number of FIFO16/RAMB16s	2	288	1%	
Number used as RAMB16s	2			
Number of DSP48s	96	96	100%	
Average Fanout of Non-Clock Nets	1.90			

Figura 4.28: Utilización final de recursos para la FPGA Virtex4

-Place and Route: se decide dónde serán colocados los componentes del diseño en el dispositivo. Dicha decisión se llevará a cabo dependiendo de si queremos optimizar en área o en velocidad.

Para culminar con éxito estos tres procesos será necesario que, además de no sobrepasar los recursos hardware disponibles en la FPGA, el diseño cumpla con las restricciones temporales (Time Constraints) que exige la implementación. La figura 4.29 muestra un sumario de estas restricciones y se verifica su cumplimiento (aparece ”Yes” en todas las filas de la columna ”Met”).

Cabe destacar que la frecuencia de reloj de la FPGA se redujo en esta implementación de 50MHz a 25MHz ya que la primera restricción temporal se incumplía.

El siguiente paso es que la suite ISE genere un bitstream (un archivo binario) de configuración de la FPGA. Este archivo servirá para que, una vez volcado en el dispositivo, este reconfigure las puertas lógicas programables dando lugar al diseño implementado en System Generator.

Una vez generado el archivo bitstream procedemos a trabajar con iMPACT. Se trata de una herramienta de ISE que permite comunicar el ordenador con la FPGA mediante una interface USB.

Cuando se abre la herramienta iMPACT aparece el la opción de seleccionar un proyecto ya existente o crear uno nuevo. Cuando creamos un proyecto nuevo tenemos distintas formas de programar el dispositivo FPGA.

Podemos programarla usando la opción ”boundary scan” en la cual se escanea una cadena de conexión y se establece una interface JTAG mediante USB que permitirá configurar la

	Met	Constraint	Check	Worst Case Slack	Best Case Achievable	Timing Errors	Timing Score
1	Yes	TS_clk_8add45c7 = PERIOD TIMEGRP "clk_8add45c7" 40 ns HIGH 50%	SETUP HOLD	0.048ns 0.986ns	39.952ns	0 0	0 0
2	Yes	COMP "gateway_out7(20)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	19.977ns	20.023ns	0	0
3	Yes	COMP "gateway_out7(19)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	20.600ns	19.400ns	0	0
4	Yes	COMP "gateway_out7(30)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	20.822ns	19.178ns	0	0
5	Yes	COMP "gateway_out7(31)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	20.823ns	19.177ns	0	0
6	Yes	COMP "gateway_out7(28)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	20.974ns	19.026ns	0	0
7	Yes	COMP "gateway_out7(27)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	20.974ns	19.026ns	0	0
8	Yes	COMP "gateway_out7(16)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.070ns	18.930ns	0	0
9	Yes	COMP "gateway_out7(29)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.083ns	18.917ns	0	0
10	Yes	COMP "gateway_out7(22)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.093ns	18.907ns	0	0
11	Yes	COMP "gateway_out7(24)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.100ns	18.900ns	0	0
12	Yes	COMP "gateway_out7(26)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.127ns	18.873ns	0	0
13	Yes	COMP "gateway_out7(21)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.198ns	18.802ns	0	0
14	Yes	COMP "gateway_out7(25)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.411ns	18.589ns	0	0
15	Yes	COMP "gateway_out7(23)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.413ns	18.587ns	0	0
16	Yes	COMP "gateway_out7(11)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.470ns	18.530ns	0	0
17	Yes	COMP "gateway_out7(13)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.590ns	18.410ns	0	0
18	Yes	COMP "gateway_out7(14)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.795ns	18.205ns	0	0
19	Yes	COMP "gateway_out7(17)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	21.934ns	18.066ns	0	0
20	Yes	COMP "gateway_out7(12)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	22.106ns	17.894ns	0	0
21	Yes	COMP "gateway_out7(18)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	22.296ns	17.704ns	0	0
22	Yes	COMP "gateway_out7(15)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	22.481ns	17.519ns	0	0
23	Yes	COMP "gateway_out7(10)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	22.739ns	17.261ns	0	0
24	Yes	COMP "gateway_out7(9)" OFFSET = OUT 40 ns AFTER COMP "clk"	MAXDE...	23.578ns	16.422ns	0	0
25	Yes	COMP "gateway_in(2)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	274.240ns	45.760ns	0	0
26	Yes	COMP "gateway_in(12)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	275.325ns	44.675ns	0	0
27	Yes	COMP "gateway_in(6)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	275.544ns	44.456ns	0	0
28	Yes	COMP "gateway_in(11)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	275.596ns	44.404ns	0	0
29	Yes	COMP "gateway_in(5)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	275.953ns	44.047ns	0	0
30	Yes	COMP "gateway_in(3)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	276.060ns	43.940ns	0	0
31	Yes	COMP "gateway_in(4)" OFFSET = IN 320 ns BEFORE COMP "clk"	SETUP	276.082ns	43.918ns	0	0

Figura 4.29: Sumario de restricciones para no sobrepasar los recursos de hardware de la PGA

FPGA con el fichero bitstream.

En la figura 4.30 podemos observar cómo se ha cargado el archivo bitstream dentro de la memoria de programación. Si se ha seleccionado la opción ”Load FPGA” se forzará a la reconfiguración automática de la FPGA cada vez que se haya cargado la memoria. La ventaja de usar la configuración mediante memoria Flash/PROM es que cuando se reinicia el dispositivo FPGA, este se reprograma automáticamente.

Cuando se ha llevado a cabo la programación de la FPGA con éxito, aparece en pantalla una etiqueta azul que indica ”Program Succeeded” y se indica que la operación ha sido exitosa en la consola de la interfaz gráfica.

Otro método igualmente válido para la programación de la FPGA es mediante una memoria Flash/PROM. La siguiente figura muestra el entorno de configuración previo a la carga de datos mediante almacenamiento Flash/PROM.

4.2.5. Estudio comparativo entre simulación software y procesado de la FPGA

La comparativa entre el modelado software y la implementación en la FPGA es importante para observar las diferencias entre el resultado obtenido trabajando mediante simulación, es un modelo ideal que no tiene en cuenta ciertos parámetros reales, y el resultado mediante procesado hardware, en el que intervienen factores que producen fluctuaciones.

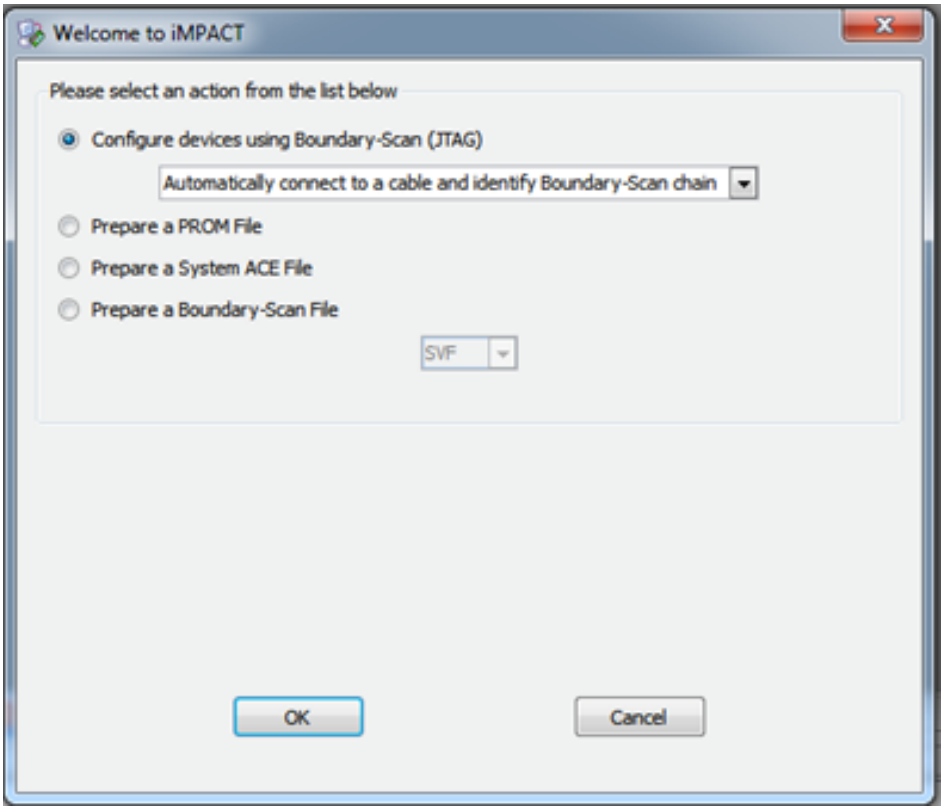


Figura 4.30: Programar la FPGA usando la opción ”boundary scan”

Para poder realizar una simulación paralela, se usará la opción de Co-simulación que ofrece System Generator. Esta opción consiste en establecer una comunicación directa entre el PC y la placa FPGA mediante la interface de programación JTAG. Dicha comunicación es independiente de la placa en la que está integrada la FPGA Virtex4 de Xilinx.

Es importante iniciar el programa desde su acceso directo. Al iniciar MATLAB directamente no se cargan algunas instancias necesarias para la comunicación entre MATLAB y la FPGA. Una vez abierto nuestro modelo, seleccionamos el bloque System Generator. En la etiqueta Çompilation"seleccionamos "Hardware Co-simulationz agregamos un nuevo dispositivo en ”New Compilation Target”.

El PIN de reloj será el P20, pues trabajaremos con el oscilador integrado en la placa. También se presentan otras opciones de trabajo como usar una señal externa de reloj o integrar otro oscilador deseado en el shocket de la placa. En la siguiente figura se puede observar el esquema de configuración del integrado ICS843001AGI-22. Este integrado se apoya en las salidas de dos osciladores de cristal y una red resistiva configurable mediante el Jumper J20 para configurar la frecuencia de su señal de salida pCLK LVPECL

Como el sistema diseñado de procesamiento de sonido no es exigente en cuanto a

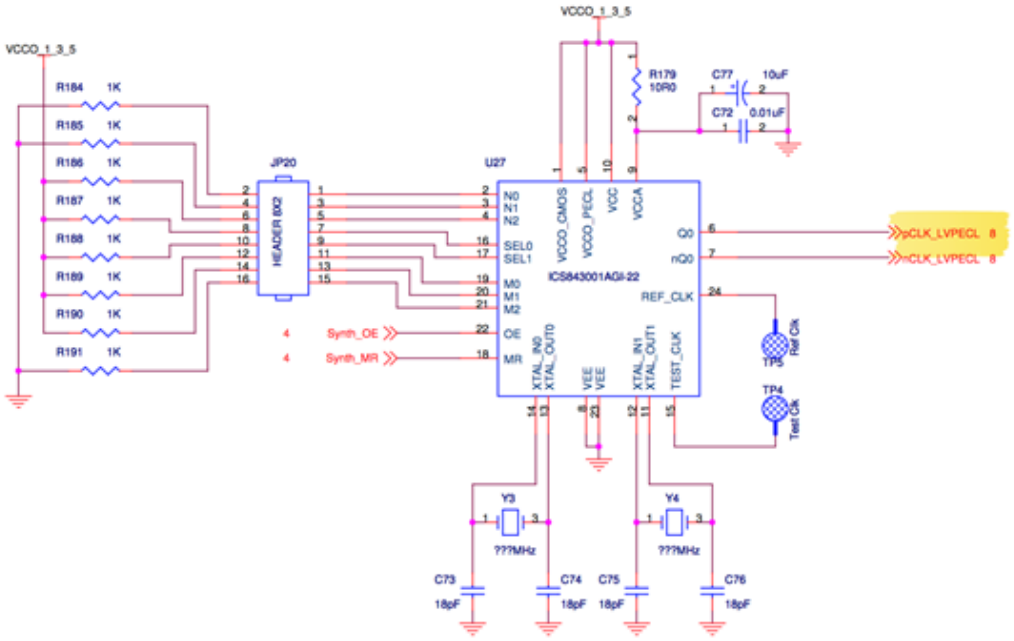


Figura 4.31: Integrado ICS843001AGI-22

frecuencia de trabajo, se opta por la configuración más baja posible. Trabajar a la menor frecuencia posible implica un menor consumo energético y una disminución de disipación de potencia. En la figura 4.32 se observa la configuración de Jumpers J20 para una frecuencia de trabajo de 74.25 MHz.

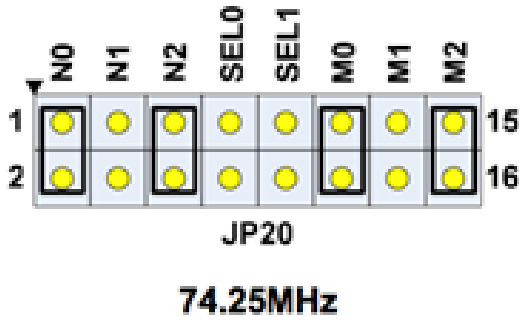


Figura 4.32: Configuración de Jumpers J20

Para localizar el pin de entrada de reloj a la FPGA nos remitimos de nuevo a la documentación de esquemáticos de la FPGA. En la figura 4.33 se observa que la salida del oscilador usado se conecta al pin P20.

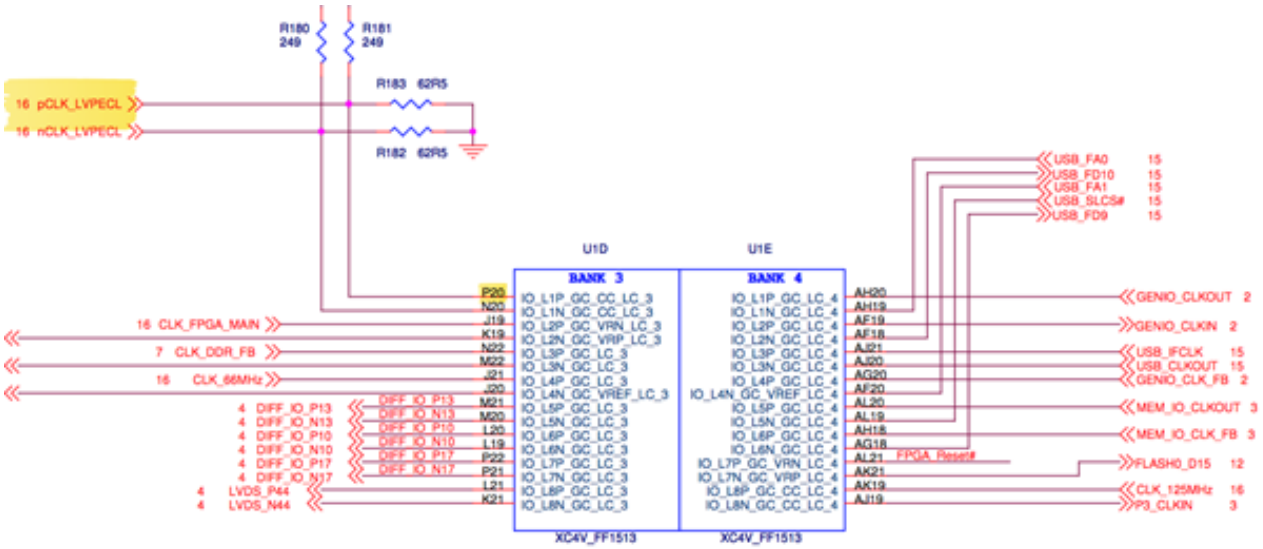


Figura 4.33: Conexión del oscilador al pin P20

Una vez realizada la configuración, se pulsa "Generate.[en] el bloque System Generator. Se realizarán los procesos de síntesis , translation, map y placeroute comentados anteriormente pero sin usar la herramienta ISE Project Navigator. Se realiza de forma transparente al usuario desde una consola integrada en el entorno de trabajo Simulink. Se crea un archivo bitstream y se configura la placa FPGA, creándose un bloque que visualizaremos como un componente de Simulink y que no es más que la interface de comunicación entre el entorno de trabajo y la FPGA.

La figura 4.34 muestra el esquemático con el bloque JTAG generado para realizar la Co-Simulación en paralelo.

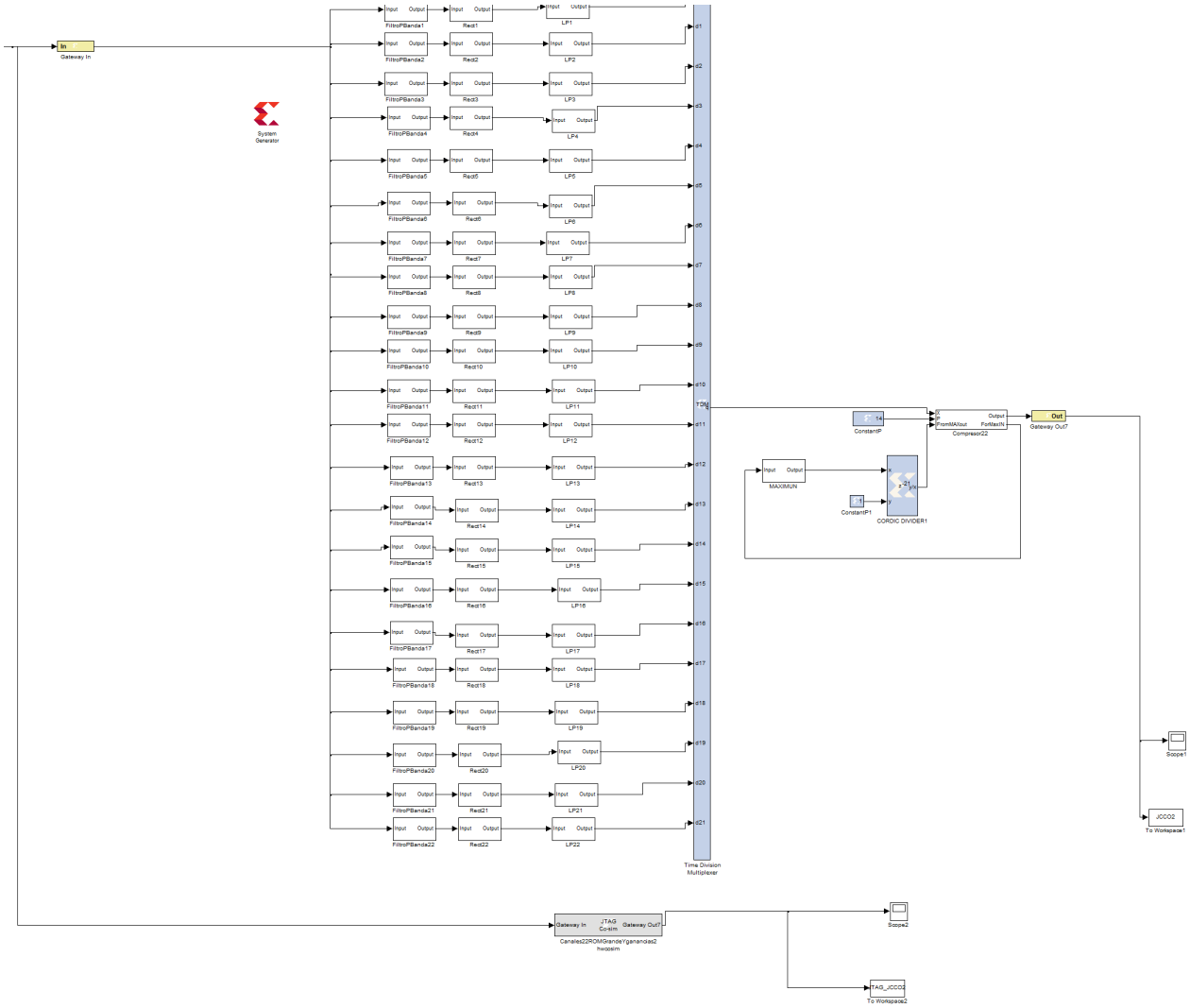


Figura 4.34: Co-Simulación en paralelo

Podemos establecer una comparativa a grandes rasgos mediante una visualización de ambas señales de salida en el dominio temporal. La figura 4.35 muestra cómo ambas señales son, a primera vista, semejantes.

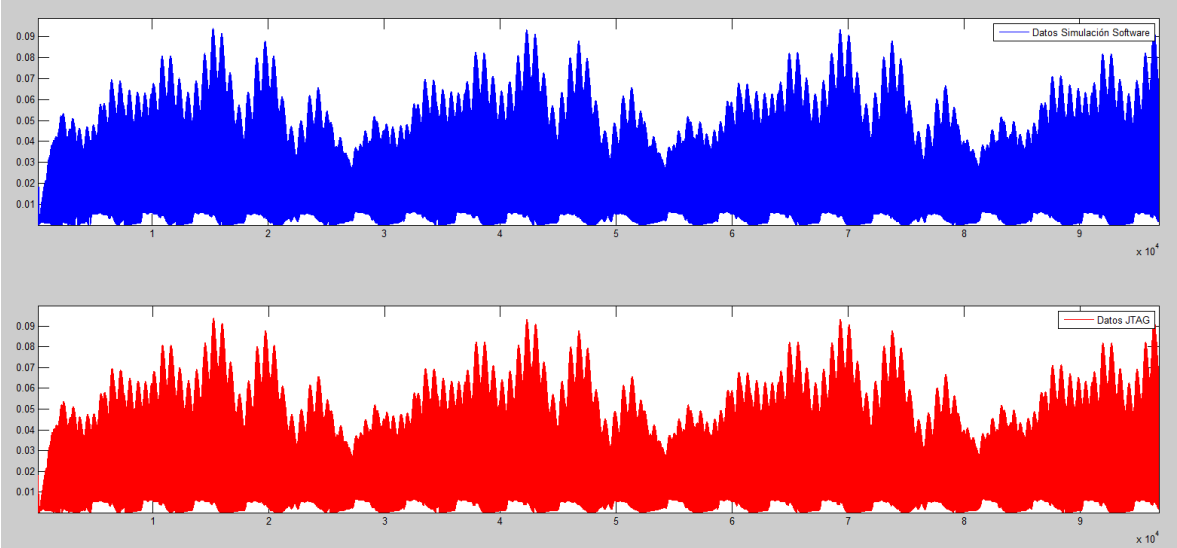


Figura 4.35: Subplot Sim vs JTAG

Si superponemos ambas representaciones y realizamos un aumento observamos cómo en realidad la señal de salida de la simulación software y la de la FPGA difieren.

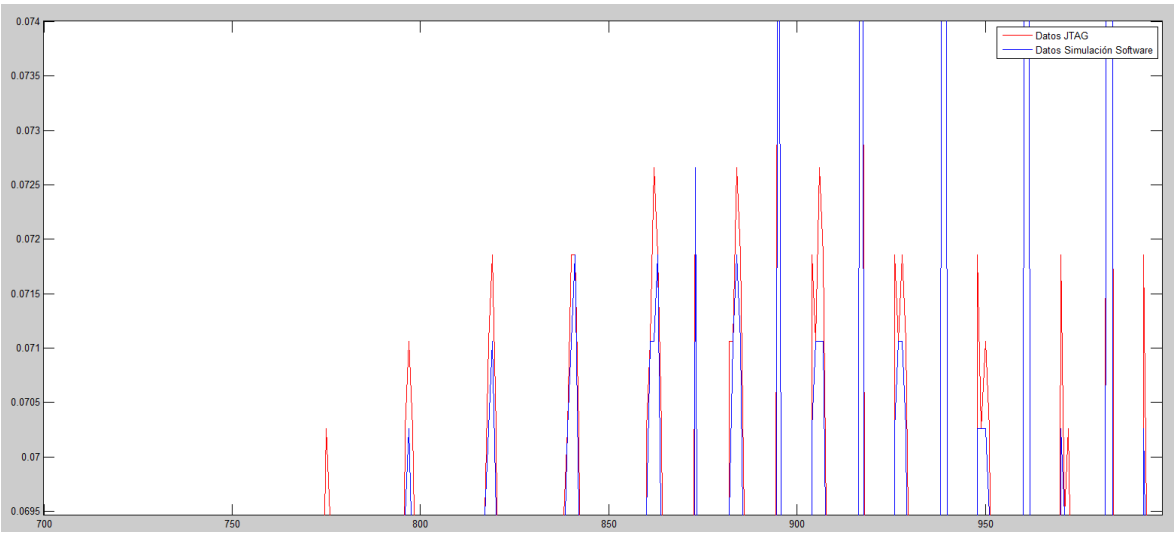


Figura 4.36: Variaciones entre JTAG y simulación software

Una vez constatada la diferencia existente entre ambas señales de salida, se procede a realizar un cálculo del error medio entre ambas para cada slot temporal. Definimos el error como el valor absoluto de la diferencia entre cada muestra multiplexada a la salida del compresor. Se usan señales de audio correspondientes a las dos primeras formantes del triángulo vocálico (A,E,O) para realizar los cálculos. En la figura 4.37 se puede observar la

evolución del error medio en cada uno de los slots de la multiplexación temporal.

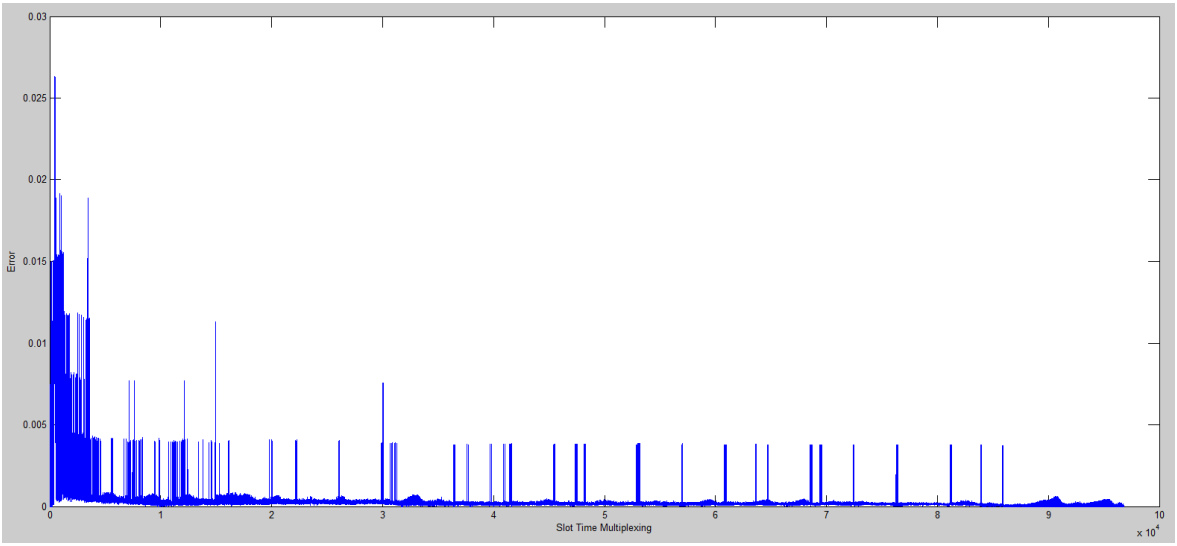


Figura 4.37: Error medio de los slots

Es necesario estudiar el error existente en cada canal de forma diferenciada ya que cada canal representa una entidad de procesamiento independiente. Mediante el Script para el cálculo de errores del ANEXO 2, podemos generar el histograma y diagrama de sectores para observar el error existente en cada canal entre la simulación hardware y la simulación software funcionando en paralelo.

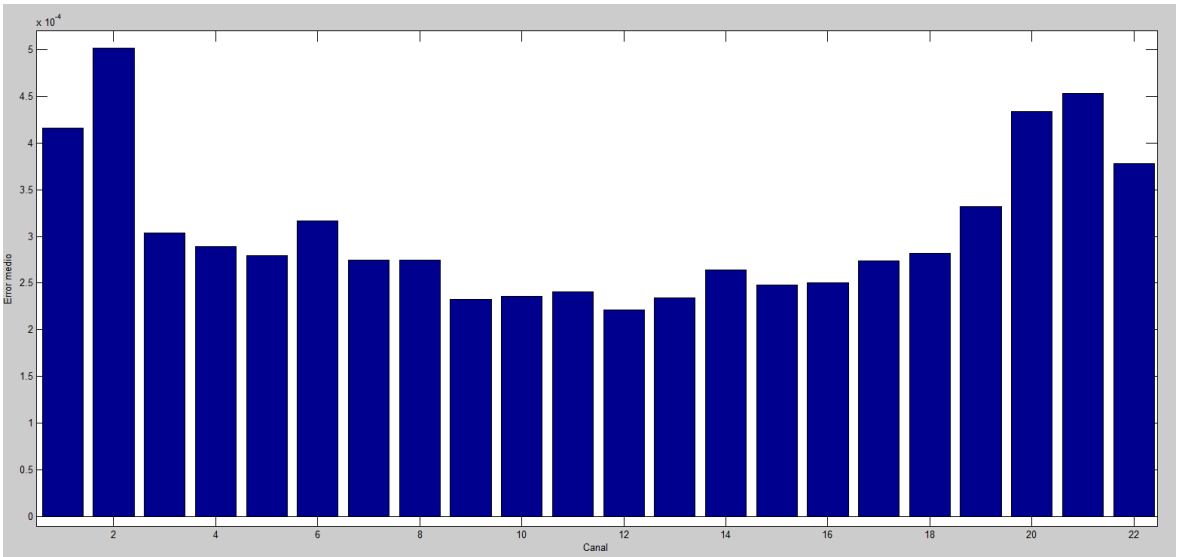


Figura 4.38: Error Medio Por Canal

En esta batería de pruebas, con las dos primeras formantes de las vocales del triángulo vocálico A,E y O, el error medio máximo es de 5.0191e-004. Teniendo en cuenta que la señal a la salida de cada canal está acotada entre 0 y 1, este valor máximo del error medio

representa un 0.0502 % del rango dinámico.

Si ahora se realiza un cálculo de la desviación típica existente en el conjunto de muestras de cada canal se obtiene el gráfico de la figura 4.39.

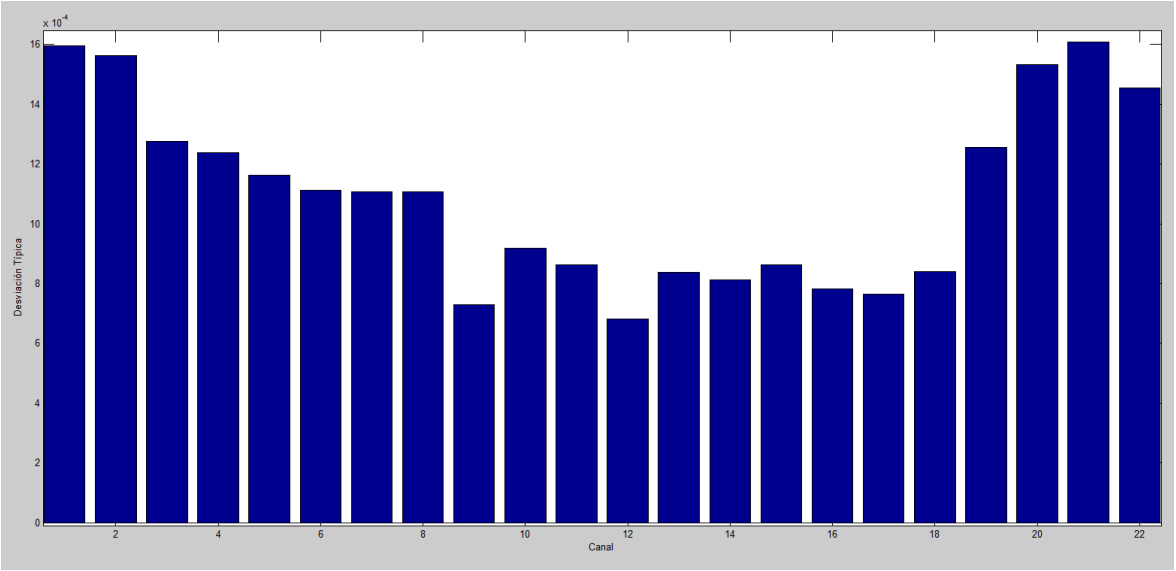


Figura 4.39: Desviacion Tipica

Se observa que la desviación típica presenta un valor muy elevado respecto al valor medio del error en cada canal. Esto indica que el cálculo realizado del error medio por canal no es representativo. Observando el cálculo del error medio por slot temporal se detecta que este es muy superior al inicio de la simulación. Esto se debe a que, hasta que el sistema alcanza un estado estacionario y se detecta al máximo global, se produce una diferencia mayor entre la señal de salida de la FPGA y la de la simulación.

Para obtener un conjunto de muestras representativas se procederá con el truncamiento de una parte inicial de las muestras procedentes de la multiplexación temporal. Truncaremos el conjunto de muestras obtenidas desde la muestra 3610, lo que supone eliminar un 3.72 % del total obtenido en la simulación.

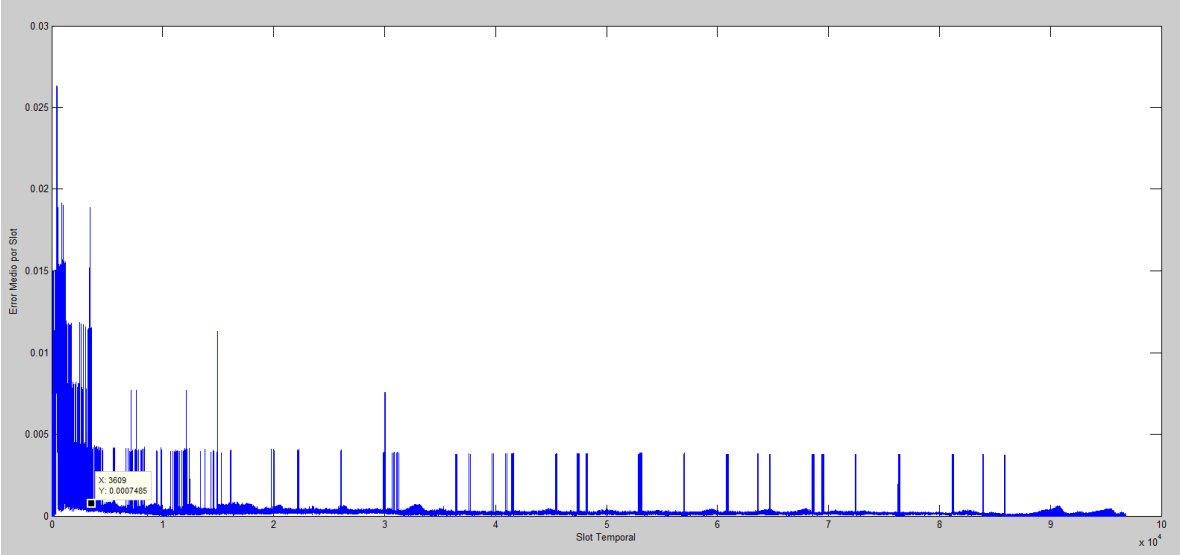


Figura 4.40: Truncamiento de las muestras

Teniendo en cuenta que el compresor produce un primer slot nulo debido al retardo que introduce de ST/22, la muestra número 3610 pertenece al canal 1 y se van multiplexando las muestras del resto de canales en los sucesivos slots. Las 164 tramas de 22 slots equivalen a 3608 slots temporales. A estos se les suma el slot nulo inicial y hacen un total de 3609 slots desechados.

En la figura 4.41 se observa el cálculo de error medio por canal con el nuevo subconjunto de muestras obtenidas. En error medio de todos los canales es 1.7338e-004, que representa un 0.017338 % del rango dinámico de la señal de salida en cada canal.

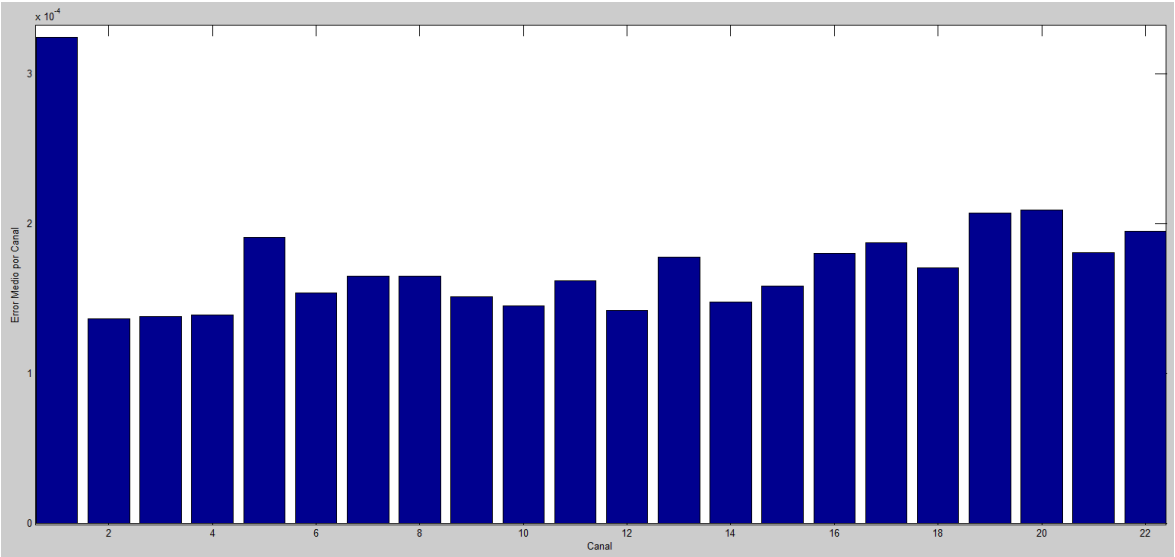


Figura 4.41: Error medio por canal

Observando la figura 4.42 se constata que la desviación típica en cada canal es

significativamente menor usando el subconjunto de muestras, indicando que la desviación respecto a la media muestral es menor y que los datos son más representativos.

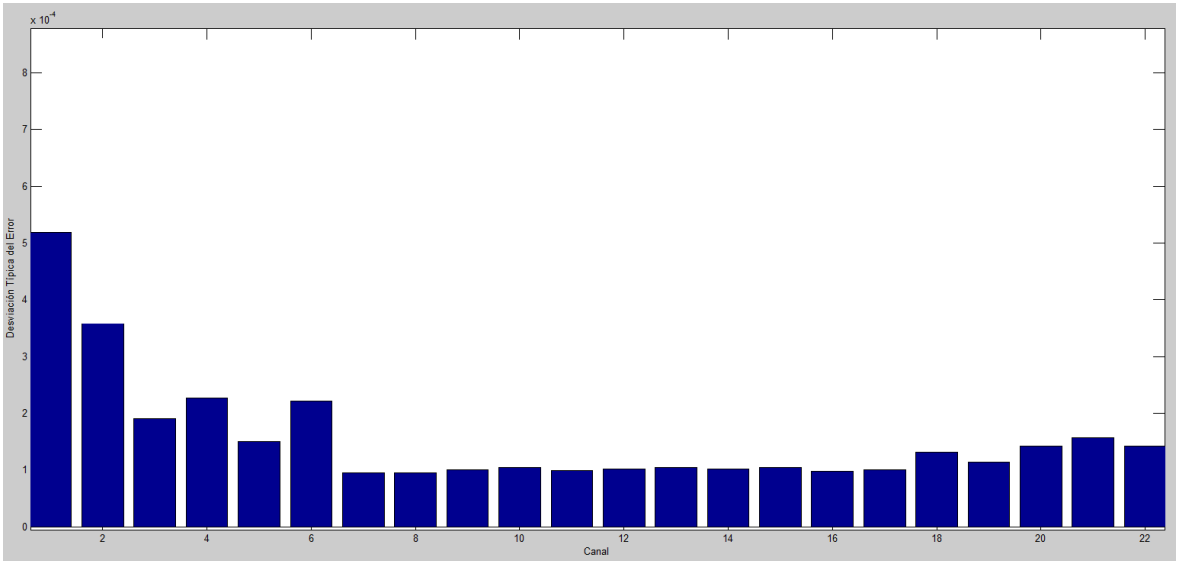


Figura 4.42: Desviación típica del error

También se puede realizar un estudio de las formantes del triángulo vocálico de la señal reconstruida. Como señales de entrada para la comparación se dispone de archivos de sonido en formato .wav pertenecientes a nueve individuos masculinos. Debido a que la población usada es pequeña, se realiza una comparación de datos estadísticos frente a otros estudios para determinar que la muestra es representativa.

A continuación se muestran las formantes de la señales de audio originales

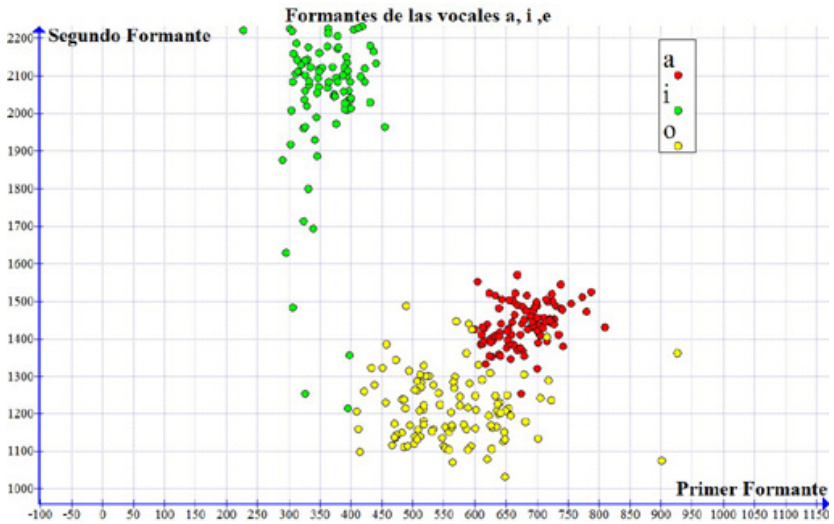


Figura 4.43: Formantes de los ficheros originales (Anexo 3 tablas C.24, C.25 y C.26)

Este gráfico (figura 4.44) representa las formantes de los ficheros procesados por la FPGA

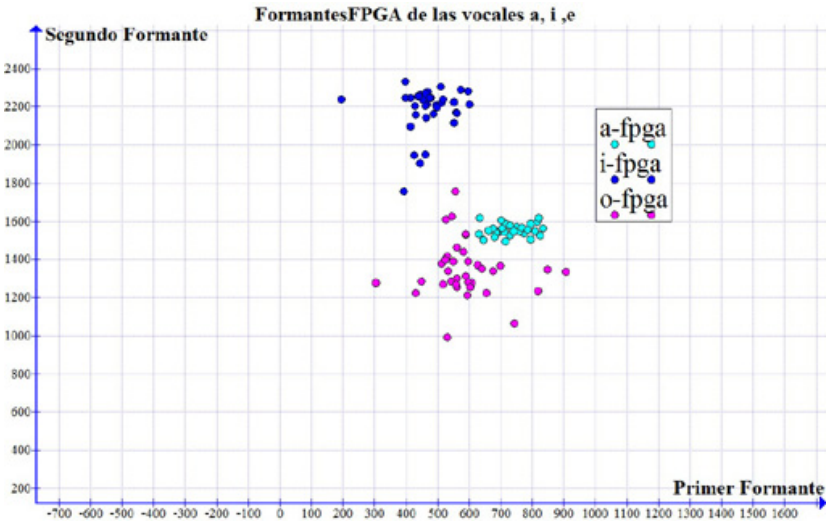
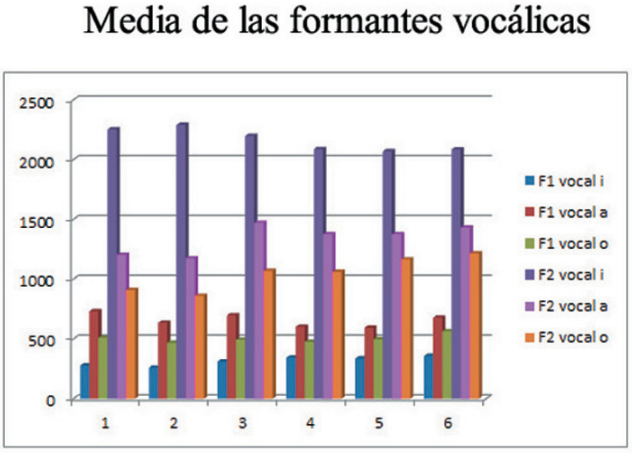


Figura 4.44: Formantes con FPGA (Anexo 3 tabla C.27)

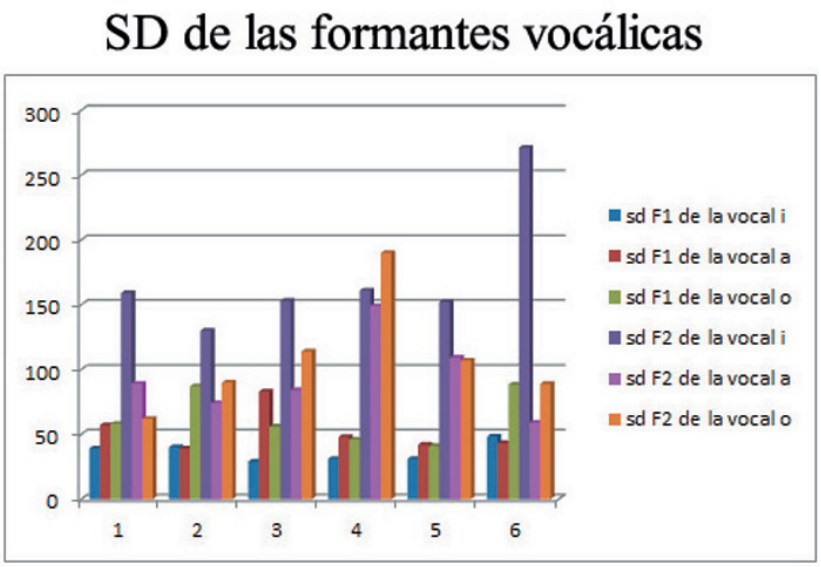
la figura 4.45 muestra una comparativa de los valores medios para cada formante del triángulo vocálico. La tabla de datos en los que se basan la comparativa siguientes se encuentra en el Anexo 3 (tablas C.24, C.25 y C.26)



[1] M. Rosique,J.L.Ramón, M. Canteras y L. Rosique [2] Quilis y Esgueva [3] Martínez Celdrán
[4]Albalá, M. J., Battaner, E., Carranza, M., Gil, J., Llisterri,J., Machuca, M. J., . . . Ríos, A
[5] Juan Julián Jiménez Gómez [6] Obtenidos por mi

Figura 4.45: Primeras y segundas formantes vocálicas

A continuación se muestran las desviaciones típicas en cada uno de los estudios



[1] M. Rosique,J.L.Ramón, M. Canteras y L. Rosique [2] Quilis y Esgueva [3] Martínez Celdrán
[4]Albalá, M. J., Battaner, E., Carranza, M., Gil, J., Llisterri,J., Machuca, M. J., . . . Ríos, A
[5] Juan Julián Jiménez Gómez [6] Obtenidos por mi

Figura 4.46: Desviación típica de las formantes vocálicas

En la figura 4.47 se visualiza el mapa de formantes del triángulo vocálico para las muestras procedentes de los nueve individuos. En este mapa de formantes se observa cierto solapamiento entre las formantes de la vocal o y de la vocal a. Para establecer que ciertamente se trata de dos clases distintas se realizará un contraste de hipótesis [13] de las medias de la primer formante de las vocales a y o, suponiendo que tienen la misma varianza, en los siguientes términos:

- μ_1 y μ_2 son respectivamente las medias de las primeras formantes de las vocales 'a' y 'o' respectivamente

- La hipótesis nula es $H_0 : \mu_1 - \mu_2 = 0$ y la alternativa es $H_1 : \mu_1 - \mu_2 \neq 0$

- El nivel de confianza es del $c = 95\,\%$, o lo que es lo mismo el nivel de significación es $\alpha = 5\,\%$

-Las muestras han de pertenecer a poblaciones normales e independientes

- Para llevar a cabo el contraste se ha utilizado el software IBM SPSS Statistics [14] versión 19 .

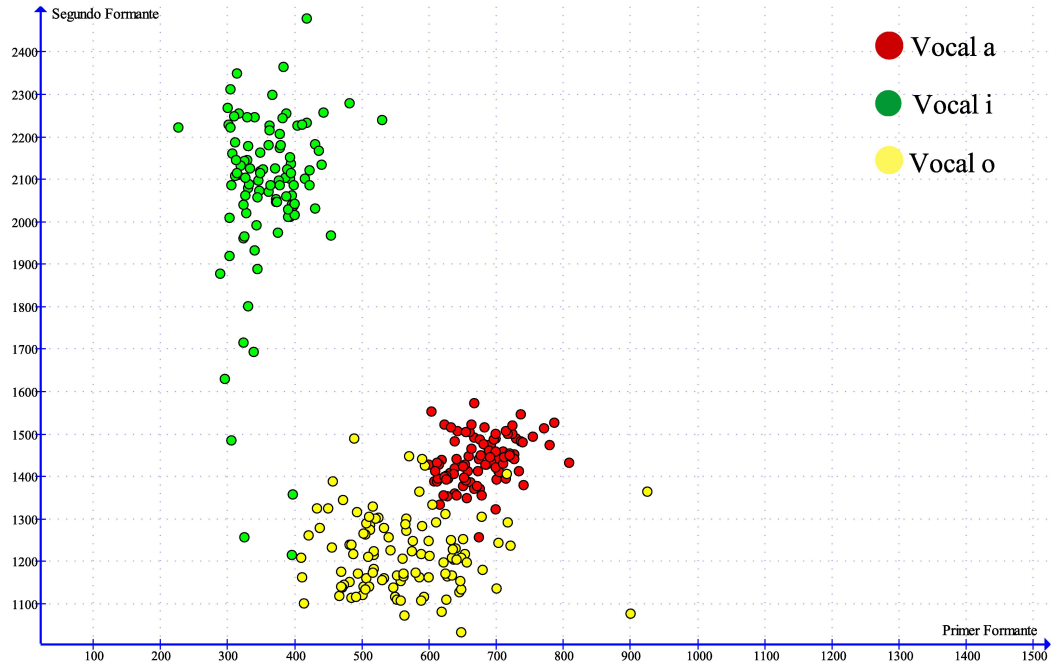


Figura 4.47: Formantes obtenidas del triángulo vocálico

En primer lugar vemos la normalidad de las primeras formantes de las vocales 'a' y 'o'. La tabla 4.1 nos da un resumen del contraste de hipótesis

Tabla 4.1: test de normalidad para la primera formante de las vocales 'a' y 'o'

Pruebas de normalidad para las primeras formantes de las vocales 'a' y 'o'						
	Kolmogorov-Smirnova			Shapiro-Wilk		
	Estadístico	gl	P-valor	Estadístico	gl	P-valor
af1	0.04	108.00	0,200	0.98	108.00	0.15
of1	0.08	108.00	0.08	0.93	108.00	0.00

El test de Kolmogorov-Smirnova se emplea para muestras de tamaño menor de 40. Para muestras con tamaño superior o igual a 40 se emplea el test de Shapiro-Wilk

La hipótesis nula es $H_0 \equiv$ la primera formante de la vocal sigue una distribución normal

La hipótesis alternativa es $H_1 \equiv$ la primera formante de la vocal no sigue una distribución normal

El P-valor para la primera formante de 'a', af1, es 0.15, superior al nivel de significación $\alpha = 0{,}05$, por tanto con una confianza del 95 % podemos afirmar que la primera formante

de la vocal a sigue una distribución normal. No ocurre esto para la primera formante de la vocal 'o', no podemos afirmar que siga una distribución normal. pero esto no es problema. En realidad no hay necesidad de comprobar la normalidad de las variables si tenemos en cuenta el Teorema Central del Límite, cuyo enunciado es el siguiente

Teorema Central del Límite.- Si una población tiene media μ y desviación típica σ, y tomamos muestras de tamaño $n > 30$, entonces las medias de estas muestras se aproximan a una distribución normal de media μ y desviación típica $\frac{\sigma}{\sqrt{n}}$

Dado el tamaño de las muestras que manejamos, no es necesario comprobar la normalidad en lo que sigue.

A continuación hacemos un test de independencia (χ^2 de Pearson) de las primeras formantes de las vocales 'a' y 'o' mediante la tabla 4.2

Tabla 4.2: Test de indepencia para las primeras formantes de las vocales 'a' y 'o'

Pruebas de chi-cuadrado para la independencia de las formantes F1a, F1o			
	Valor	grados de libertad	P-valor
Chi-cuadrado de Pearson	10.152.00	10.058.00	0.25
Razón de verosimilitudes	974.25	10.058.00	1.00
Asociación lineal por lineal	0.33	1.00	0.56
N de casos válidos	108.00		

Al ser el $P - valor = 0{,}25 > \alpha = 0{,}05$, con una confianza del 95 % podemos afirmar que las primeras formantes de las vocales 'a' y 'o' son independientes.

Por último procedemos con el contraste de hipótesis de la igualdad de medias anteriormente mencionado. En el software IBM SPSS ejecutamos el test comparar medias/ Prueba T para muestras independientes, obteniendo los resultados que refleja las tablas 4.3 y 4.4:

1º Test de Levene para la igualdad de varianzas

Tabla 4.3: Test de Levene para contrastar las varianzas de las primeras formantes de 'a' y 'o'

	Prueba de Levene para la igualdad de varianzas	
	F(Snedecor)	P-valor
Se han asumido varianzas iguales	31.01	0.00
No se han asumido varianzas iguales		

Al ser el $P-valor = 0{,}00 < \alpha = 0{,}05$ se rechaza la hipótesis nula, por tanto las varinzas

de las primeras formantes de las vocales 'a' y 'o' son distintas

Tabla 4.4: Contraste de medias para la primera formante de 'a' y 'o'

	Prueba T para la igualdad de medias					
	t-Student	gl	P-valor	Diferencia de medias	Intervalo de confianza para diferencia de medias Inferior	Superior
varianza iguales	12.15	214	0.00	114.92	96.28	133.56
varianza distintas	12.15	155,44*	0.00	114.92	96.24	133.60

*Los grados de libertad para un coeficiente parcial determinado se basan en el número menor de casos utilizado en el cálculo de dicho coeficiente. Los decimales que aparecen en los gl. se deben a que el software IBM SPSS utiliza, en el caso de varianzas distintas, la siguiente expresión (ecuación de Welch) para estimar los grados de libertad:

$$gl = \frac{\left(\frac{S_1^2}{n_1} + \frac{S_2^2}{n_2}\right)^2}{\left(\frac{S_1^2}{n_1 - 1}\right)^2 + \left(\frac{S_2^2}{n_2 - 1}\right)^2}$$

siendo S_1 y S_2 las varianzas, respectivamente, de la primera y segunda muestras, n_1 y n_2 el número de datos de cada muestra. .

El P-valor para varianzas distintas es 0.00, por tanto se rechaza la hipótesis nula y podemos afirmar, con un 95 % de confianza, que las primeras formantes de las vocales 'a' y 'o' son distintas.

Procedemos de la misma manera para contrastar la hipótesis de que la media de las segundas formantes de las vocales 'a' y 'o' son distintas. Sin entrar en más detalles ponemos los resultados obtenidos con SPSS:

Tabla 4.5: test de normalidad para la segunda formante de las vocales 'a' y 'o'

Pruebas de normalidad de las segundas formantes de 'a' y 'o'

vocal	Kolmogorov-Smirnova			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
a	0.05	108.00	,200*	0.99	108.00	0.41
o	0.10	108.00	0.02	0.97	108.00	0.01

Tabla 4.6: test de independencia para la segunda formante de las vocales 'a' y 'o'

Pruebas de chi-cuadrado de independencia 'af2' y 'of2'			
	Valor	gl	Sig. asintótica (bilateral)
Chi-cuadrado de Pearson	216,00	203.00	0.25
Razón de verosimilitudes	299.44	203.00	0.00
Asociación lineal por lineal	144.18	1.00	0.00
N de casos válidos	216.00		

Tabla 4.7: Test para comprobar la homocedasticidad de la segunda formante de 'a' y 'o'

		Prueba de Levene para la igualdad de varianzas	
		F	Sig.
f2	Se han asumido varianzas iguales	15.770	.000
	No se han asumido varianzas iguales		

Tabla 4.8: Contraste de igualdad de medias para la segunda formante de 'a' y 'o'

	Prueba T para la igualdad de medias					
	t	gl	P-valor	Diferencia de medias	Intervalo de confianza para la diferencia	
					Inferior	Superior
varianzas iguales	20.87	214.00	0.00	214.21	193.98	234.44
varianzas distintas	20.87	185,77*	0.00	214.21	193.96	234.45

*Los grados de libertad para un coeficiente parcial determinado se basan en el número menor de casos utilizado en el cálculo de dicho coeficiente.

Si observamos la tabla 4.8, al ser el P-valor cero, podemos concluir que con una confianza del 95 % las medias de las segundas formantes de 'a' y 'o' son distintas.

Resumiendo, los solapamientos que hemos observado en las gráficas al representar las formantes primera frente a la segunda en las vocales 'a', y 'o' no han de preocuparnos ya que se trata de poblaciones distintas (con una confianza del 95 %).

Según se puede ver en el gráfico 4.48, volvemos a tener un problema de solapamiento entre las formantes de la vocal 'a' y las de 'o' con FPGA

Hacemos un contraste de hipótesis para descartar que las poblaciones de las primeras formantes F1a y F1-oFPGA son distintas. También hacemos el mismo test para descartar que las poblaciones de las segundas formantes F2a y F2-oFPGA coincidan. La explicación de los contrastes que vamos a efectuar son exactamente los mismos que los que hemos llevado a cabo en el anterior solapamiento de las vocales 'a' y 'o' sin FPGA de modo que tan solo

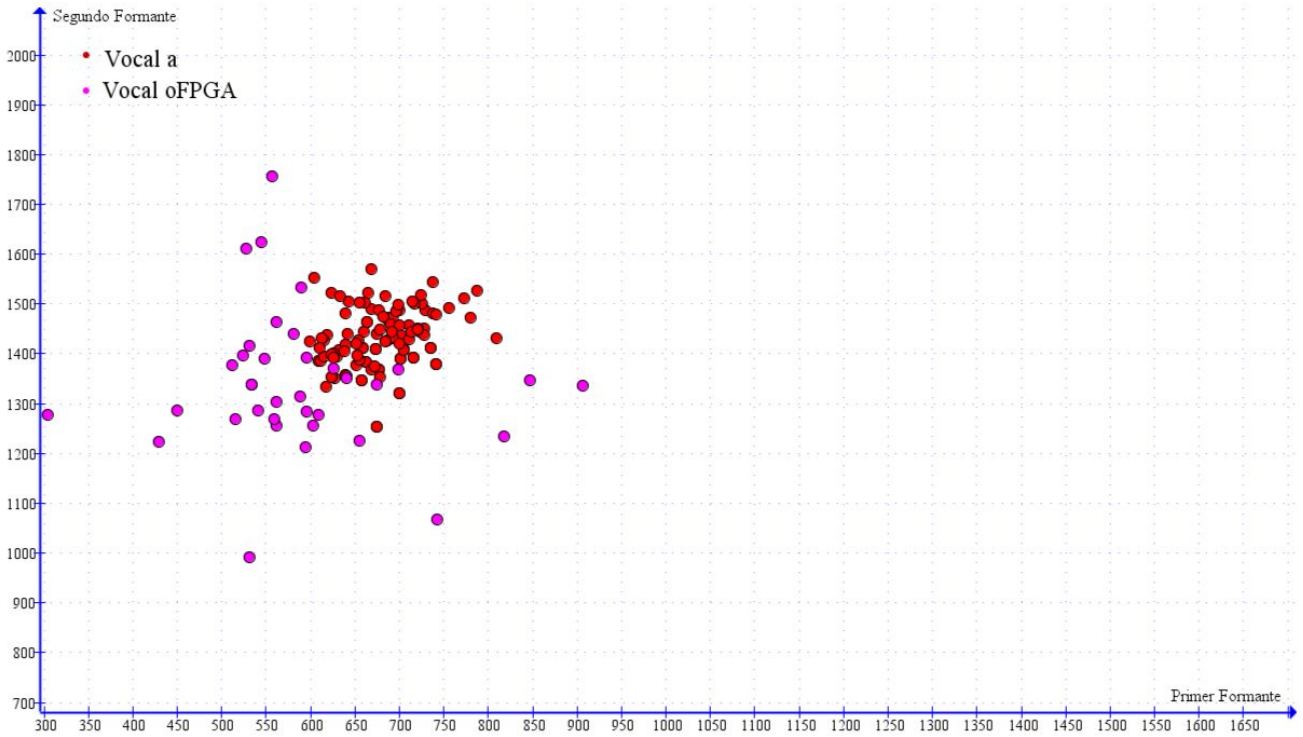


Figura 4.48: Formantes vocálicas de 'a' y 'o-FPGA'

presentaremos los resultados

a) Primeras formantes para las vocales 'a' y 'o'- con FPGA

En la tabla 4.9 se observa que la primera formante de la vocal 'a' sigue una distribución gausiana, no siendo así en el caso de 'o-FPGA' pero al ser el tamaño de la muestra $n = 36 > 30$ el Teorema Central del Límite nos asegura que podemos aproximar 'o-FPGA' a una normal.

La tabla 4.10 nos muestra un P-valor para la χ^2 de Pearson de 0.17, lo que nos lleva a la conclusión, con el 95 % de confianza, la primera formante de la vocal 'a' es independiente de la 'o'-FPGA

El test de Levene que muestra la tabla 4.11 nos dice que las varianzas de 'a' y 'o'-FPGA son distintas, lo necesitaremos para contrastar las medias mediante la t-Student

El contraste de las medias de la primera formante de las vocales 'a' y 'o'-FPGA de la tabla 4.12 muestra que, con la debida reserva del 95 % de confianza, las medias son distintas.

Tabla 4.9: Test de normalidad para las primeras formantes de 'a' y 'o-FPGA'

vocal	Kolmogorov-Smirnova			Shapiro-Wilk		
	Estadístico	gl	P-valor	Estadístico	gl	P-valor
a	0.04	108.00	0.20	0.98	108.00	0.15
o-FPGA	0.18	36.00	0.01	0.90	36.00	0.00

Tabla 4.10: Test de independencia de la primera formante de 'a' y 'o-FPGA'

Pruebas de chi-cuadrado para la indepencia de 'a' y 'o-FPGA'			
	Valor	grados de libertad	P-valor
Chi-cuadrado de Pearson	144.000.00	129.00	0.17
Razón de verosimilitudes	161.95	129.00	0.03
Asociación lineal por lineal	37.03	1.00	0.00
N de casos válidos	144.00		

Tabla 4.11: Contraste de igualdad de varianza para la primera formante de 'a' y 'o-FPGA'

	Prueba de Levene para la igualdad de varianzas	
	F	P-valor
Se han asumido varianzas iguales	21.46	0.00
No se han asumido varianzas iguales		

Tabla 4.12: Contraste de igualdad de medias para primera formante de 'a' y 'o'-FPGA

	Prueba T para la igualdad de medias						
	t	gl	P-valor	Diferencia de medias	Error típ. de la diferencia	Intervalo de confianza para la diferencia de medias	
						Inferior	Superior
Varianzas iguales	7.04	142.00	0.00	90.97	12.91	65.45	116.50
Varianzas distintas	4.75	38,52*	0.00	90.97	19.14	52.25	129.70

*Los grados de libertad para un coeficiente parcial determinado se basan en el número menor de casos utilizado en el cálculo de dicho coeficiente.

b) Segundas formantes para las vocales 'a' y 'o'- con FPGA

Tabla 4.13: test de normalidad para la segunda formante de 'a' y 'o'-FPGA

	Kolmogorov-Smirnova			Shapiro-Wilk		
	Estadístico	gl	P-Valor	Estadístico	gl	P-Valor
a	0.05	108.00	0.20	0.99	108.00	0.41
o-FPGA	0.15	36.00	0.04	0.93	36.00	0.02

Tabla 4.14: Prueba de independencia de segunda formante de 'a' y 'o'-FPGA

Pruebas de chi-cuadrado para la segunda formante de 'a' y 'oFPGA'			
	Valor	gl	Sig. asintótica (bilateral)
Chi-cuadrado de Pearson	144.00	130.00	0.19
Razón de verosimilitudes	161.95	130.00	0.03
Asociación lineal por lineal	25.03	1.00	0.00
N de casos válidos	144.00		

Tabla 4.15: Prueba de homocedasticidad para la segunda formante de 'a' y 'o'-FPGA

	Prueba de Levene para la igualdad de varianzas	
	F	P-valor
Se han asumido varianzas iguales	19.51	0.00
No se han asumido varianzas iguales		

Tabla 4.16: Contraste de medias de la segunda formante de 'a' y 'o'-FPGA

	Prueba T para la igualdad de medias						
	t	gl	P-valor	Diferencia de medias	Error típ. de la diferencia	Intervalo de confianza para la diferencia de medias Inferior	Superior
varianzas iguales	5.489	142	.000	91.7964980	16.7241463	58.7360218	124.8569742
varianzas distintas	3.784	39.117	.001	91.7964980	24.2589275	42.7328827	140.8601133

Los contrastes para la segunda formante de las vocales 'a' y 'o' con PGA nos indican que ambas poblaciones tienen distinta media.

Resumiendo, los solapamientos anteriormente mencionados no han de preocuparnos, no hay posibilidad de confusión al tratarse de poblaciones distintas (ya que tienen distinta media) todo ello al 95 % de confianza.

A continuación damos un resumen de estadística descriptiva de las muestras que hemos manejado en las formantes vocálicas

Si comparamos los datos que refleja la estadística descriptiva de mis muestras (ANEXO 3) con los de la tabla 4.17 de formantes vocálicas en español según distintos autores [12] observamos que:

1º Las medias de las muestras que he utilizado en las formantes de las vocales 'a', 'i' y 'o' sin FPGA no difieren mucho de las que obtuvieron los mencionados autores. Véase la siguiente tabla

Tabla 4.17: Media y desv.típica de las formantes vocálicas en Hz. según diversos autores

		[1]		[2]		[3]		[4]		[5]	
	Vocal	Media	Sd	Media	sd	Media	sd	Media	sd	Media	sd
	[i]	278	39	260	40	313	29	345	31	338	31
	[e]	465	44	411	59	457	40	432	33	436	34
F1	[a]	733	57	636	39	699	83	603	48	594	42
	[o]	510	58	469	87	495	56	477	46	499	41
	[u]	306	61	272	51	349	38	-	-	362	33
	[i]	2254	159	2291	130	2200	153	2089	161	2071	152
	[e]	1972	123	2019	113	1926	117	1764	193	1732	135
F2	[a]	1203	89	1175	74	1471	84	1375	149	1375	109
	[o]	909	62	862	90	1070	114	1062	190	1166	107
	[u]	752	127	671	66	877	128			983	99

Valores medios y desviaciones típicas para F1 y F2 de las vocales en los estudios de [1] M. Rosique, J.L.Ramón, M. Canteras y L. Rosique (1979), [2] Quilis y Esgueva (1983), [3]Martínez Celdrán (1995), [4] Albalá M.J., Battaner E., Carranza M., Gil J., y otros (2008)

, [5] Juan Julián Jiménez Gómez

2º Es curioso que según los datos de los distintos autores y mis muestras, las segundas formantes se aproximan al doble de la frecuencia de las primeras formantes en los casos de las vocales 'a' e 'i' y en el caso de la vocal 'o' la segunda formante viene a ser el cuádruple de la primera. Esto queda representado en las siguientes gráficas.

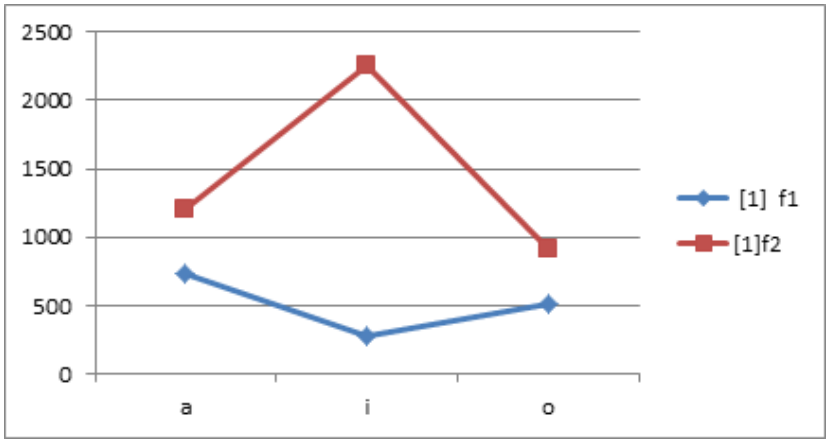


Figura 4.49: Formantes según M. Rosique, J.L.Ramón, M. Canteras y L. Rosique

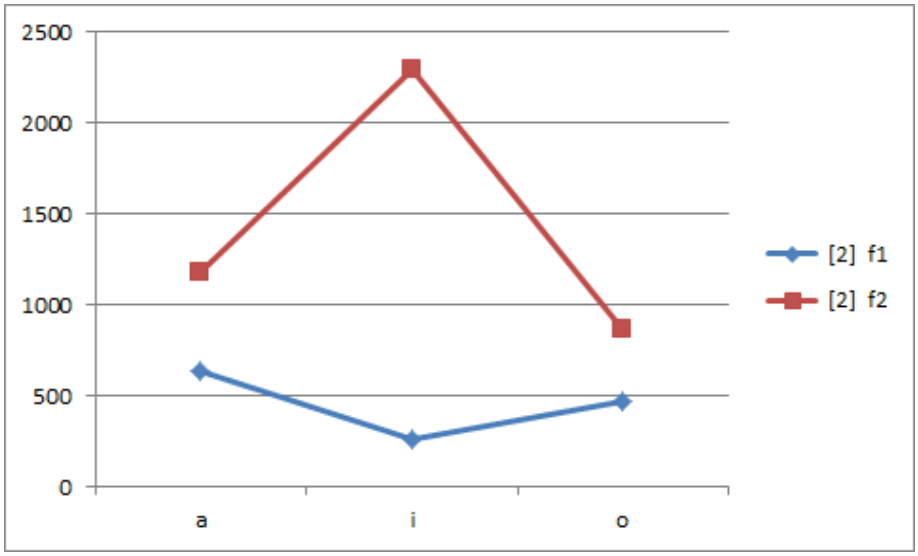


Figura 4.50: Formantes según Quilis y Esgueva

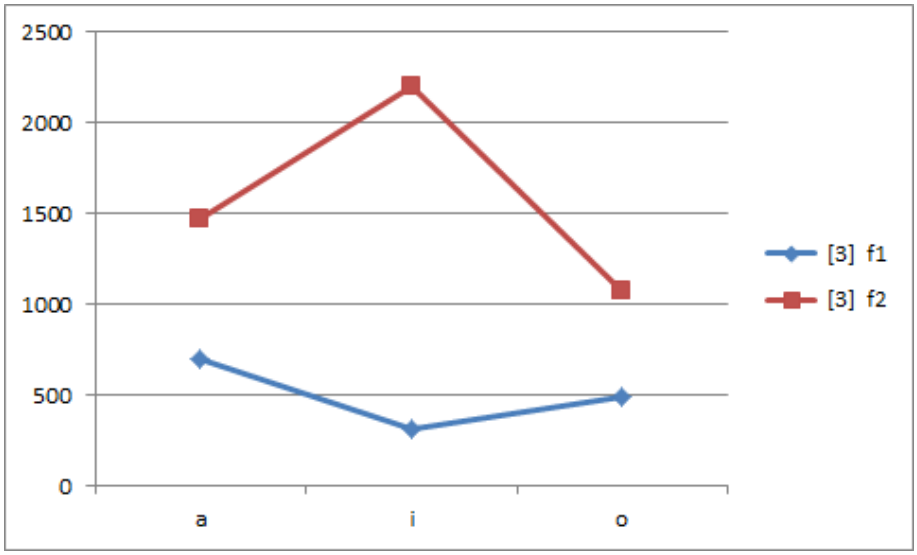


Figura 4.51: Formantes según Martínez Celdrán

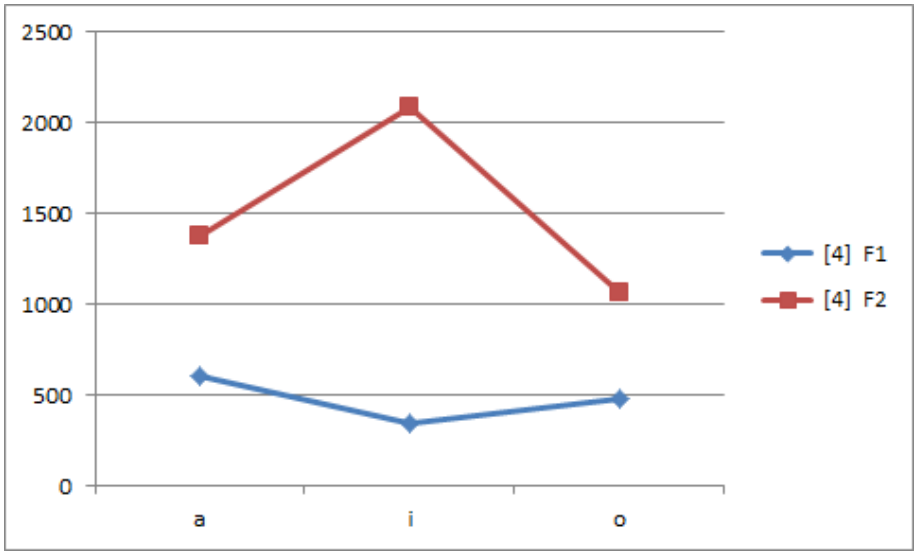


Figura 4.52: Formantes según Albalá M.J., Battaner E., Carranza M., Gil J., y otros

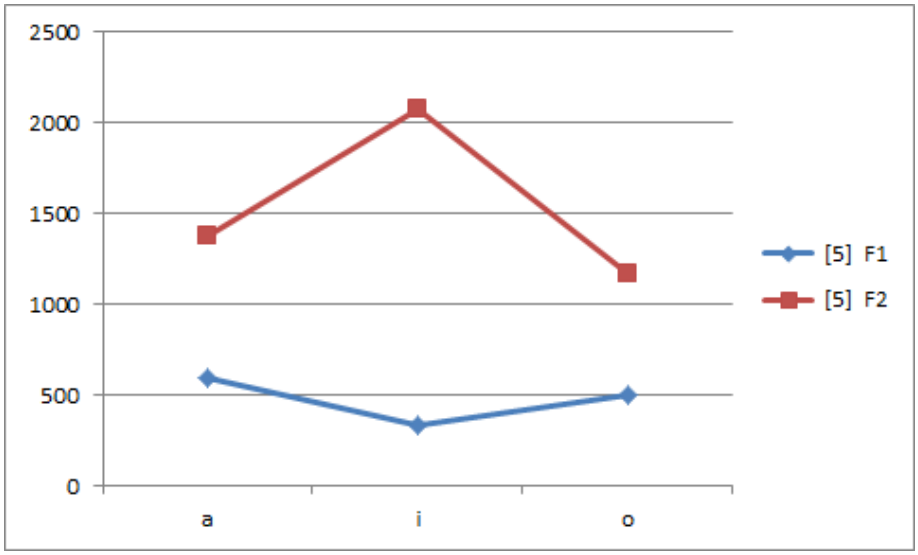


Figura 4.53: Formantes según Juan Julián Jiménez Gómez

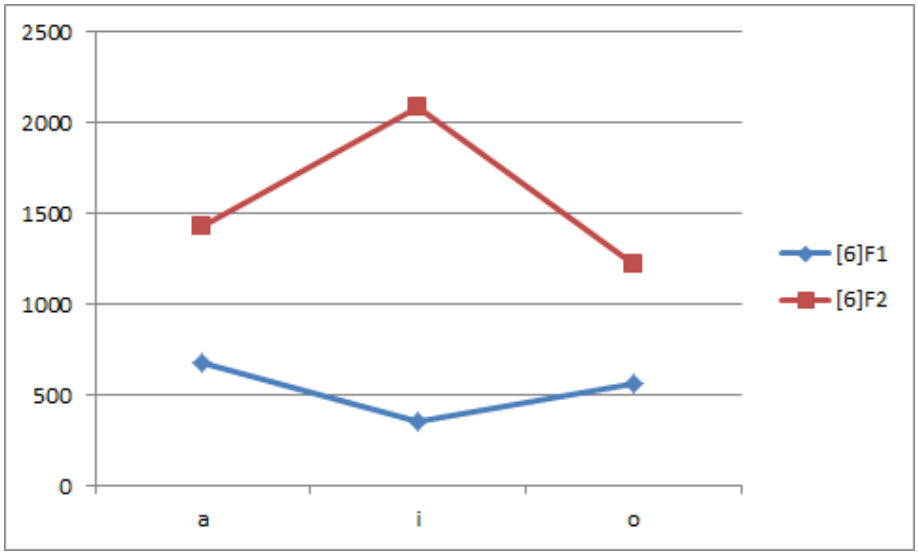


Figura 4.54: Formantes según mis muestras

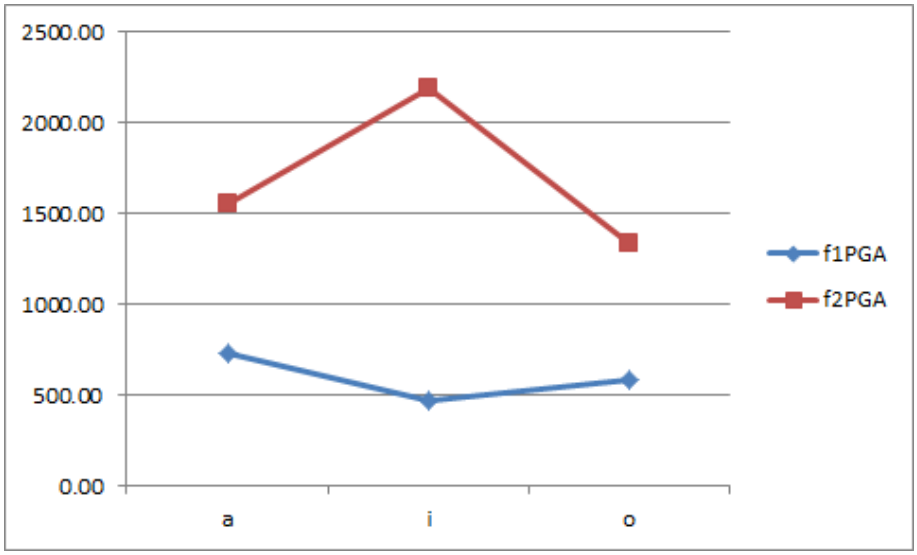


Figura 4.55: Formantes PGA según mis muestras

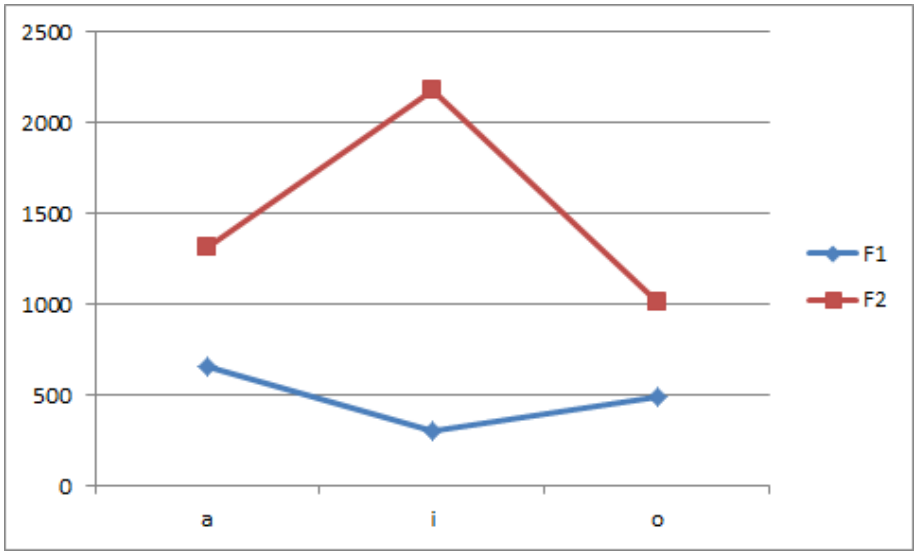


Figura 4.56: Formantes del conjunto de distintos autores

Una vez estudiado mediante el triángulo vocálico el funcionamiento del sistema implementado, para tener una visión más global del procesamiento llevado a cabo, se analizarán los resultados del mapa vocálico completo. Para ello se usará como entrada al sistema un conjunto de muestras procedentes de diez individuos. De cada uno de ellos se obtendrán las cinco vocales.

En la figura 4.57 se muestra el mapa de formantes de los ficheros de voz originales (tabla C.26 del Anexo 3), antes de ser procesados.

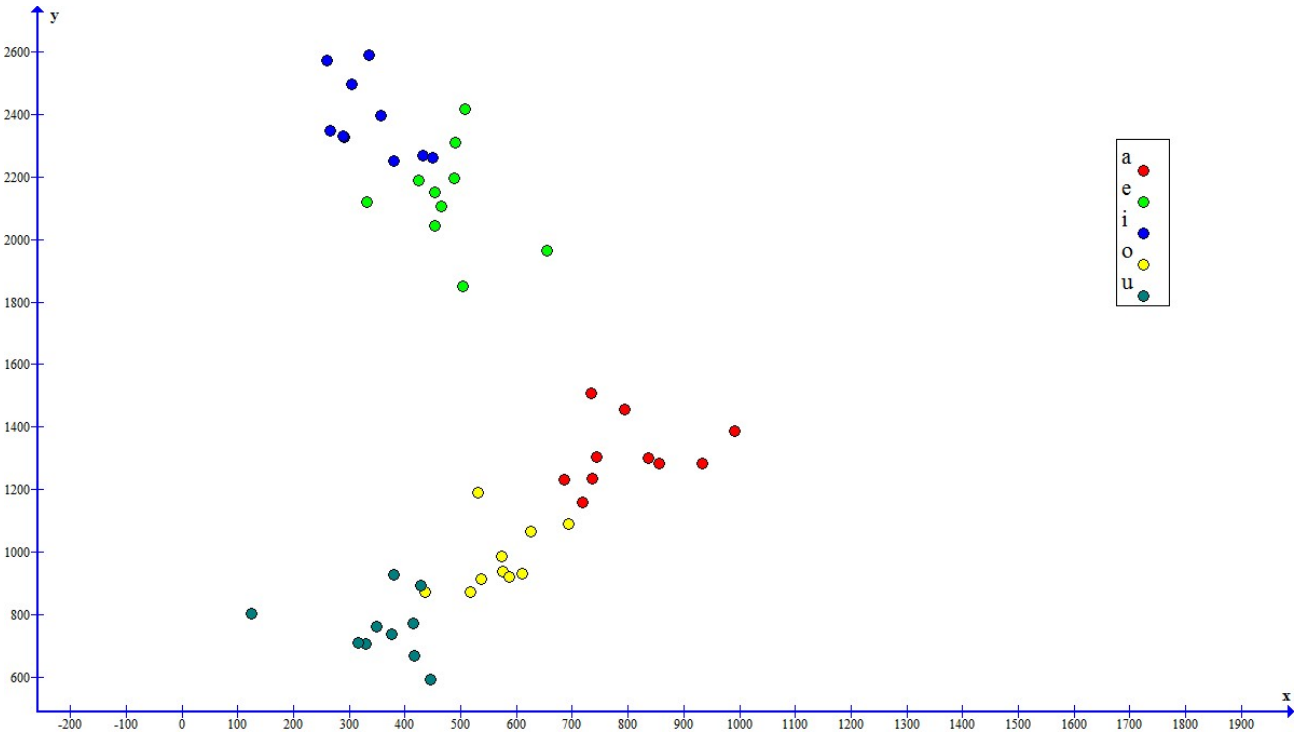


Figura 4.57: Formantes originales

Una vez procesados estos ficheros de sonido, se observa en la figura 4.58 un desplazamiento generalizado de las formantes hacia frecuencias más altas (tabla C.27 del Anexo 3).

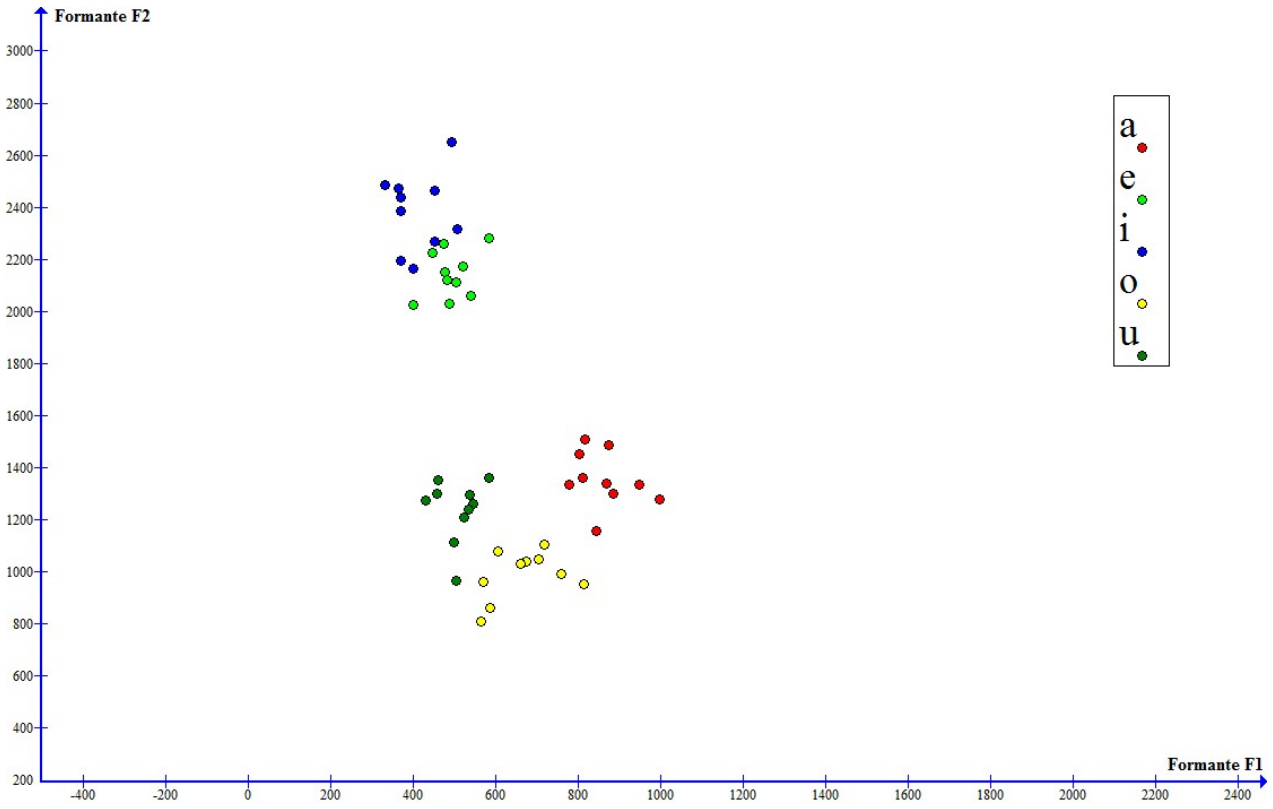


Figura 4.58: Formantes después del procesado

En la figura 4.59 se observa como se han desplazado las medias de las formantes vocálicas después del procesado

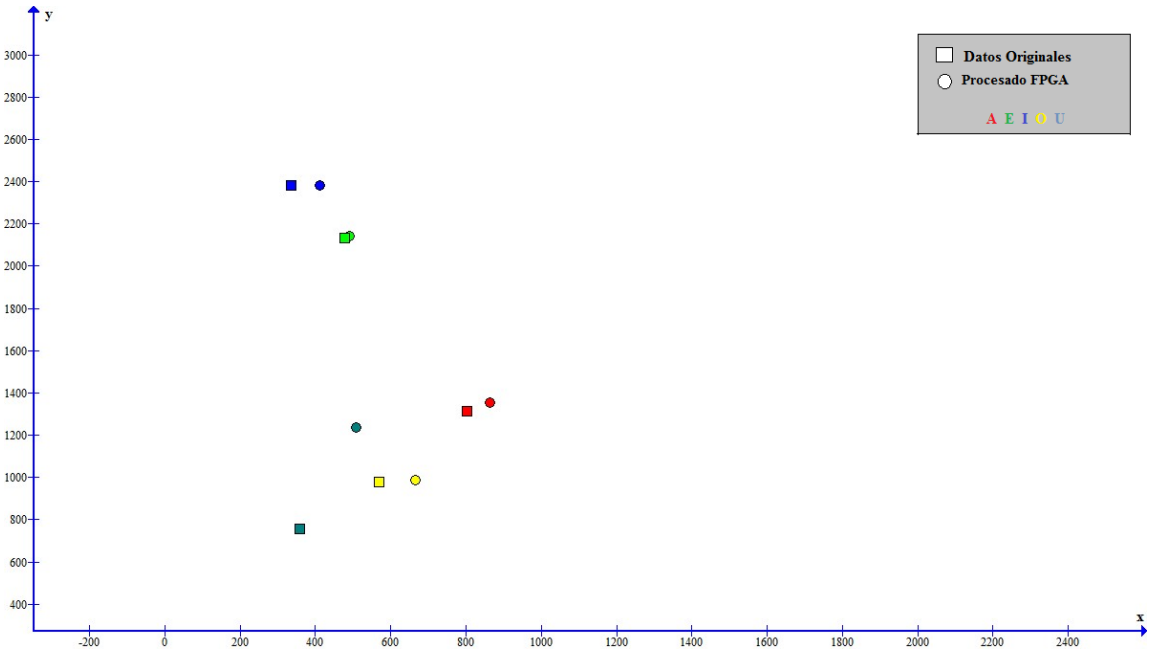


Figura 4.59: Desplazamiento de las medias de las formantes

El desplazamiento de las formantes se deriva del método de reconstrucción usado después del procesado. El hecho de que las formantes vocálicas se desplacen en frecuencia indica que la persona implantada percibe el habla de forma distinta a una persona normoyente. Esto no es tan relevante a la hora de establecer un parámetro cualitativo para determinar el éxito del diseño. Sí lo es el hecho de que las formantes se agrupen en regiones determinadas del mapa de formantes sin presentar un solapamiento frecuente y significativo.

Capítulo 5

Conclusiones

En este Proyecto Fin de Carrera se ha llevado a cabo una labor de documentación con la cual se ha podido tener una visión global y aproximada al mundo de la implantación con fines auditivos. Después de adquirir unos conocimientos iniciales, se comenzó a desarrollar la estrategia de codificación del habla CIS, estudiando el comportamiento de cada etapa de procesado del conjunto de veintidós canales y relacionando cada bloque con un comportamiento biológico y psicoacústico.

Una vez diseñado e implementado el sistema en un modelo hardware, se ha observado el comportamiento global del sistema de codificación tomando como referencia las formantes vocálicas antes y después del procesado. Las formantes han sufrido un desplazamiento a frecuencias superiores. Sin embargo el sistema implementado ha seguido realizando una clasificación de las vocales en conjuntos disjuntos, con lo que se siguen pudiendo establecer fronteras de decisión entre clases. Es decir, se mantienen las diferencias vocálicas sin presentar solapamientos significativos de formantes que dificulten la distinción.

Se han generado ficheros de audio mediante la reconstrucción de la señal procesada. El resultado observado es una constatación de que, dependiendo del grado de daño en el sistema auditivo, la percepción auditiva del implantado difiere de la de un normoyente.

La estrategia CIS que se ha tratado no es más que una pequeña muestra del conjunto de técnicas de codificación del habla usadas actualmente. Multitud de equipos de investigación estudian día a día cómo mejorar estas estrategias y crear nuevos métodos de codificación que se adapten mejor al complejo mundo de la psicoacústica.

La implantación auditiva abarca estrategias multidisciplinares de las áreas biomédica y electrónica. Gracias a la evolución de estos dispositivos, hoy en día se consigue superar la barrera social y cognitiva que supone la imposibilidad de comunicarse verbalmente de muchas personas. La evolución de los implantes cocleares va de la mano de los últimos avances en electrónica. Los implantes cocleares son dispositivos cada vez más miniaturizados, más precisos y con mayor autonomía.

Las investigaciones para tratar la hipoacusia severa avanzan en dos caminos. Por un lado el perfeccionamiento de las técnicas de codificación del habla y el uso de implantes ipsilaterales, los cuales estimulan las células ciliadas remanentes con el fin de potenciar la audición y acercar la percepción del implantado a la de un normoyente. Por otro lado se están llevando a cabo investigaciones con células madre, con el objetivo de regenerar las células ciliadas y las células del ganglio espiral mediante el uso de neurotrofinas y factores de crecimiento.

Entendiendo las telecomunicaciones como el conjunto de áreas científicas que estudian las técnicas para llevar a cabo comunicaciones a larga distancia, se puede asegurar que no existe mayor distancia para la comunicación que la imposibilidad de escuchar a otra persona. Es por todo esto que los implantes cocleares y el resto de técnicas que ayudan a superar barreras cognitivas deben permanecer como prioridades indiscutibles en las áreas de investigación.

Anexo A

Anexo 1

A.1. ANOTACIONES SOBRE EL USO DE LOS MÓDULOS DE SYSTEM GENERATOR TOOLBOX

El uso de ésta herramienta requiere un periodo de adaptación y, aunque la curva de aprendizaje es mucho más rápida que el lenguaje VHDL, precisa comprender algunos conceptos concernientes a su uso. En este apartado se detallarán incidencias que se han dado a la hora del diseño y simulación para que puedan ser de ayuda para futuros proyectos o trabajos realizados con System Generator. También se explicarán los bloques usados en el diseño, tanto los bloques System Generator como los bloques Simulink.

A.1.1. Bloque SYSTEM GENERATOR

Podría definirse como el corazón que vertebra el diseño hardware en System Generator. Será necesario usar el bloque System Generator siempre que nos dispongamos a realizar un diseño hardware en Simulink . En éste bloque se determinaran parámetros de reloj de la FPGA, parámetros temporales de la simulación en aritmética flotante y opciones de implementación.

Si no se usa este bloque habiendo usado componentes de la Toolbox System Generator, se producirá automáticamente un error fácilmente detectable.

Sólo es necesario el uso de un bloque System Generator en todo el diseño. Si usamos más de uno los iconos que se muestran son simplemente espejos de un único bloque ya que el diseño es síncrono y en ningún caso podrán establecerse distintas frecuencias de reloj en el mismo diseño.

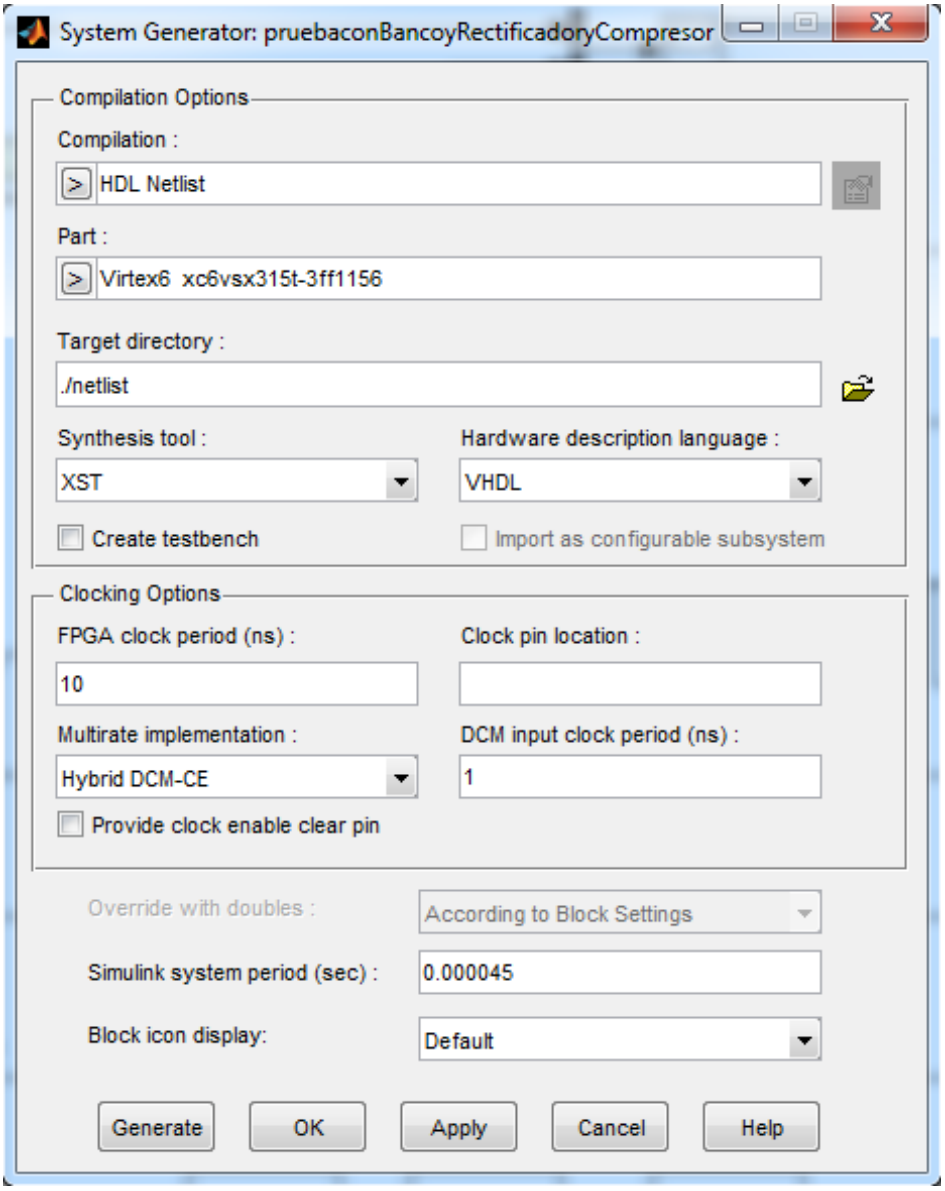


Figura A.1: Prueba con banco/rectificador/compresor

A.1.2. GATEWAYS

El usuario de System Generator debe tener claro desde un principio que en el entorno de diseño se van a trabajar con dos modos diferenciados. El diseño Simulink trabajará con aritmética flotante y el diseño hardware trabajará en aritmética fija. Es por esto que cada vez que se realice un cambio de frontera entre ambos entornos deberemos usar los bloques GATEWAY.

En términos de electrónica podrían definirse como conversores analógico-digitales y digital-analógicos.

El bloque Gateway in realiza un muestreo a una frecuencia especificada por el usuario (deberá ser un múltiplo de la frecuencia de reloj establecida en el bloque System Generator). También nos ofrece la opción de determinar el tipo de redondeo para la cuantificación y la configuración de bits su salida (número de bits para la parte entera y fraccionaria).

El bloque Gateway out transforma los tipos de Sistem Generator a tipos Simulink. Podemos seleccionar en opciones sus restricciones temporales, traducción a puerto de salida o los recursos hardware de la FPGA.

En este proyecto se han usado los parámetros de configuración por defecto de este bloque.

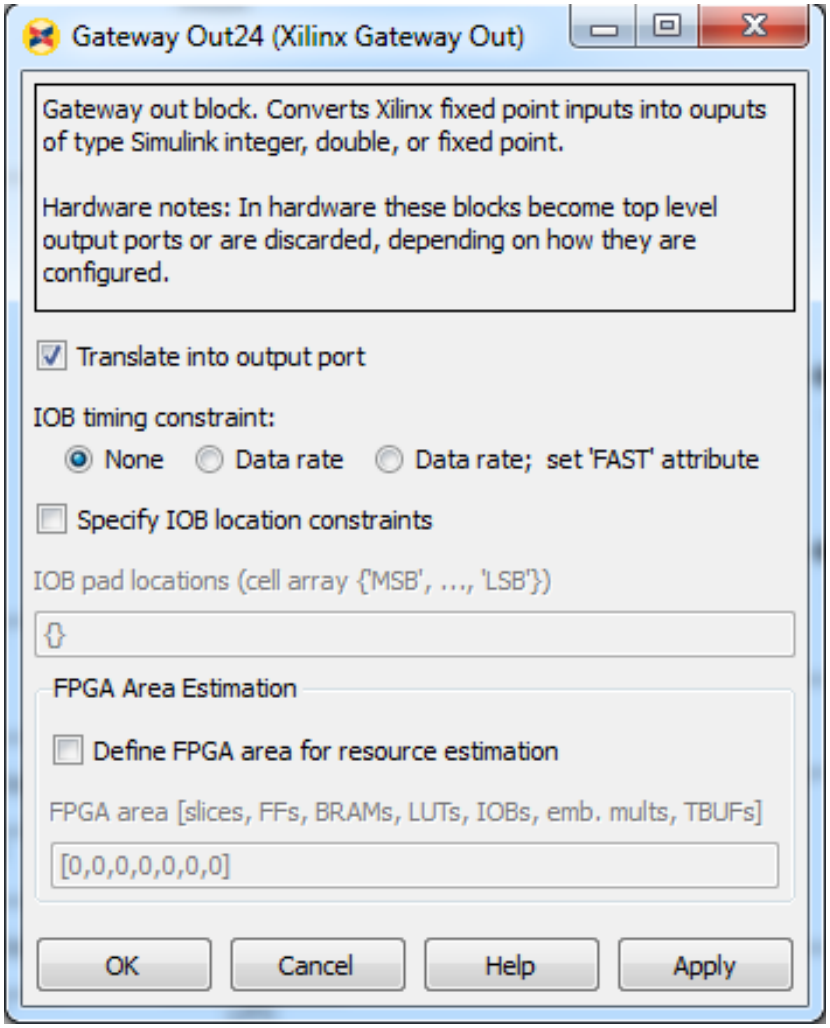


Figura A.2: El bloque Gateway out transforma los tipos de Sistem Generator a tipos Simulink

A.1.3. CONSTANT

Éste bloque es esencial en el diseño de este proyecto fin de carrera debido al uso de filtros que requieren parámetros. Dichos parámetros son números decimales muy pequeños y el principal problema que puede surgir es no especificar bien el tipo de datos.

En el caso de que no se especifiquen bien el número de bits para la parte real y para la fraccionaria se produce un error al redondear la cifra y, debido a los valores pequeños con los que se trabajan, esto no será admisible.

Este bloque nos permite cargar automáticamente datos que se encuentran en el workspace de MATLAB. Esto facilita la tarea cuando dichos coeficientes son calculados mediante una herramienta MATLAB como FDATool.

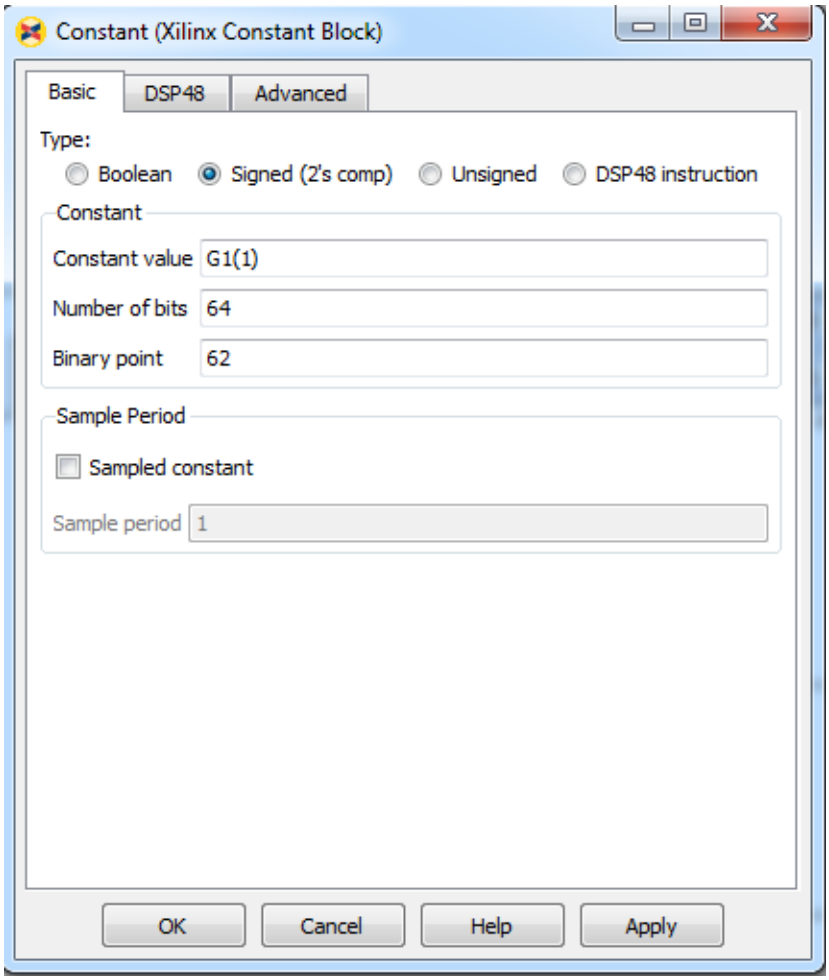


Figura A.3: El bloque Constant nos permite cargar automáticamente datos que se encuentran en el workspace de MATLAB

A.1.4. MEMORIAS ROM

El principal problema que se ha dado en este proyecto a la hora de usar dichas memorias fue en el bloque del Compresor . El acceso a memoria requiere un retardo de, como mínimo, un ciclo de reloj. Cuando estamos operando con un valor de un bus previo a dicha memoria y el resultado de lectura de la misma será necesario introducir un retardo para que exista una coherencia temporal entre ambos operandos.

Es por esto que en el Compresor se usa un registro que provoca un retardo de un ciclo de reloj en la entrada al compresor. En ningún caso podremos seleccionar un retardo nulo en una memoria, ya que es una restricción que presenta el diseño hardware.

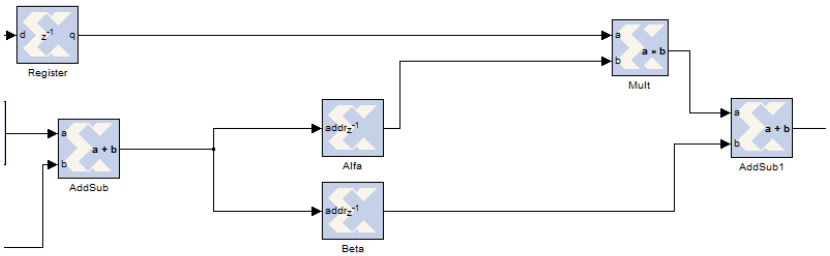


Figura A.4: No es posible un retardo nulo en una memoria, ya que es una restricción que presenta el diseño hardware.

A.1.5. ASSERT

Se ha requerido el uso de este bloque en el diseño de los módulos MAXIMUM y MINIMUN dentro del sistema Compresor. Este bloque permite la intervención directa del usuario en el tipo de datos del bus y las tasas de muestreo. En un principio no se tuvo en cuenta esta restricción a la hora del diseño de ambos sistemas lo cual produjo error a la hora de simular.

Siempre que se requiera un sistema retroalimentado será necesario el uso de este bloque.

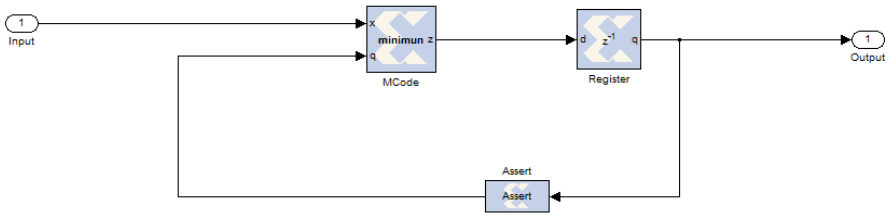


Figura A.5: Siempre que se requiera un sistema retroalimentado será necesario el uso del bloque Assert

A.1.6. MCode

En un principio este bloque puede parecer una panacea para el usuario de System Generator, ya que el poder usar un lenguaje con nivel de abstracción superior como es el de MATLAB abre un gran abanico de posibilidades, sin embargo no es así del todo. Este bloque únicamente permite el uso de estructuras de selección. Es por esto que no podremos usar cualquier función de MATLAB.

Una cosa muy importante a tener en cuenta es que los parámetros de entrada a la función que implementa este bloque estarán en artmética fija pero no los números que usemos al implementar la función, que estarán en flotante. Es por esto que tendremos que realizar una conversión (casting) de los números al formato fijo con xfix.

A.1.7. CONVERT

Este bloque es usado en el sistema rectificador. La primera duda que se plantea al usar este bloque es si realiza un cambio en el valor del flujo de bits del bus. La respuesta es negativa, este bloque solo realiza un CAST, que consiste en una reinterpretación de dicho flujo.

Por ejemplo si en el bus se encuentra una secuencia en complemento a dos y lo reinterpretamos como sin-signo, el bit de signo de la palabra en C2 pasará a ser el bit más significativo de la palabra sin-signo.

A.1.8. CORDIC DIVIDER

Este módulo implementa el algoritmo matemático CORDIC, originalmente presentado por Volder en 1959 para el cálculo de funciones trigonométricas. El 1974 Walther modificó dicho algoritmo para calcular funciones elementales (multiplicación y división) , funciones hiperbólicas y logarítmicas [11].

La implementación hardware de dicho algoritmo para el cálculo de funciones elementales presenta ventajas respecto al ahorro de componentes hardware. Dicho algoritmo es realizable con registros de desplazamiento y sumadores (shifts and add implementation).

CORDIC es un algoritmo iterativo, se basa en tres ecuaciones iterativas. Dichas ecuaciones son las siguientes:

$$x_{k+1} = x_k - m\delta_K y_k 2^{-k}$$
$$y_{k+1} = y_k + \delta_K x_k 2^{-k}$$
$$z_{k+1} = z_k - \delta_K \sigma_k$$

Las constantes m, δ_k y σ_k dependen del cálculo específico que se esté realizando como se explica a continuación:

1. Los posibles valores de m son 0, 1 ó -1. m=1 se utiliza cuando se trabaja con funciones trigonométricas e inversas de trigonométricas. m=-1 se usa para funciones hiperbólicas y sus inversas, funciones exponenciales y logarítmicas y raíces cuadradas. Por último m=0 se utiliza para efectuar productos y cocientes que es nuestro objetivo.

2. δ_k es una de las dos funciones de signo que se dan a continuación:

$$\delta_k = sgn(z_k) = \begin{cases} 1 & z_k \geq 0 \\ \text{-1} & z_k < 0 \end{cases}$$

ó

$$\delta_k = -sgn(y_k) = \begin{cases} 1 & y_k < 0 \\ \text{-1} & y_k \geq 0 \end{cases}$$

La primera es conocida con el nombre técnico de ”rotation mode”, para valores de z que convergen a cero, y se usa en el algoritmo para multiplicar. La segunda se conoce técnicamente con el nombre de ”vectoring mode”, para valores de y que convergen a cero y se utiliza en el algoritmo de la división.

3. los valores σ_k dependen de m. Si m=0 entonces $\sigma_k = 2^k$. Para m=1 $\sigma_k = tan^{-1}2^{-k}$ y si m=-1 vale $\sigma_k = tanh^{-1}2^{-k}$

Para aplicar el algoritmo CORDIC se ha de partir del conocimiento de los valores iniciales apropiados x_0, y_0 y z_0. Estos valores iniciales se deben restringir a un entorno de cero de manera que quede garantizada su convergencia.

En nuestro caso, para realizar la división $\frac{x_0}{y_0}$ las ecuaciones son estas:

$$\begin{aligned} x_{k+1} &= x_0 \\ y_{k+1} &= y_k + \delta_K x_k 2^{-k} \\ z_{k+1} &= z_k - \delta_K \sigma_k \end{aligned}$$

En este caso $m = 0$ y $\delta_k = -sgn(y_k)$. Los valores iniciales son x_0, y_0, tomando el valor $z_0 = 0$

El hecho de ser un algoritmo iterativo hace que se presenten restricciones temporales cuando va a ser usado en serie. La acumulación de retardos podría incumplir los requerimientos de velocidad del diseño.

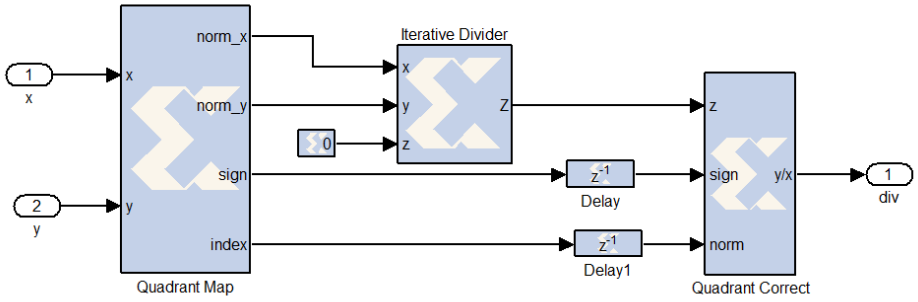


Figura A.6: módulos que forman el bloque CORDIC DIVIDER

En la siguiente Figura podemos observar los módulos que forman el bloque CORDIC DIVIDER.

El algoritmo de división se realiza en cuatro pasos:

-Paso 1: Coordinate Rotation. El algoritmo CORDIC sólo converge para valores positivos de X. Si X es negativo, se trasladará el vector de entrada al cuadrante más cercano, en sentido antihorario, tal que X tenga un valor positivo.

-Paso 2: Normalización. Este algoritmo sólo converge para valores de y>=2x. Para valores de x>y, tanto x como y sufren un desplazamiento hacia la izquierda tal que el bit más significativo sea 1. El desplazamiento relativo de y respecto a x se pasa al bloque Quadrant Correct.

-Paso 3: Rotación Lineal. Para el cálculo del cociente, el vector se desplaza en ángulos pequeños hasta que y tiende a cero, como se especifica en las ecuaciones iterativas para la división.

-Paso 4: Co-ordinate Correction. Mediante el último bloque del sistema se realiza una corrección de la rotación del valor x y del desplazamiento relativo de y respecto x.

La siguiente figura muestra el bloque Co-ordinate Correction. El multiplexor realiza el cambio de cuadrante y se multiplica el resultado por $2^{desplazamiento\ relativo\ de\ y\ sobre\ x}$.

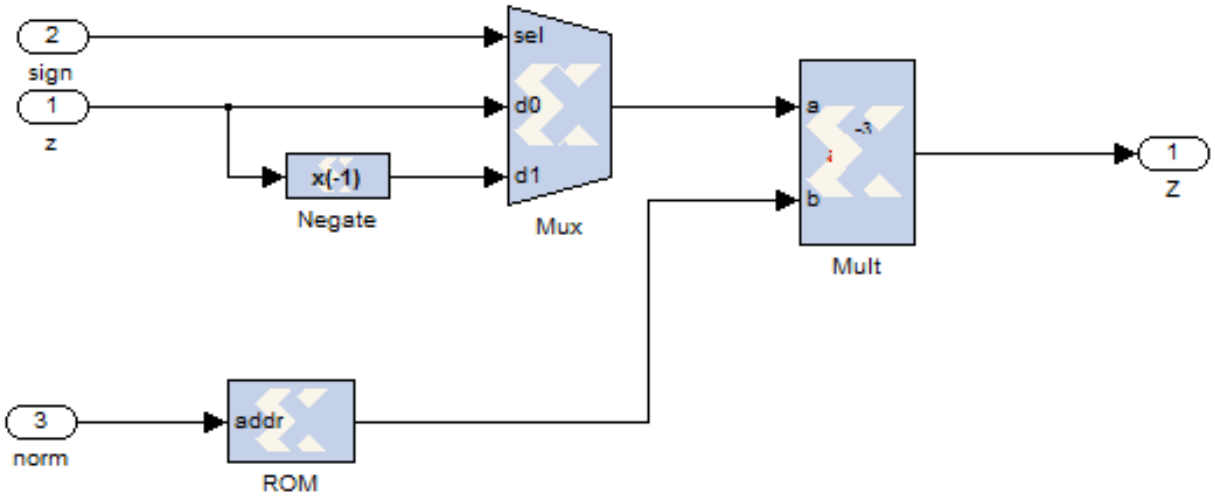


Figura A.7: Bloque Co-ordinate Correction

A.2. Bloques de Simulink

A.2.1. Signal Generator

Este bloque nos permite generar señales senoidales, cuadradas, de diente de sierra y aleatorias, nos servirá para generar una señal a medida para la prueba inicial del diseño de filtros. En este Proyecto Fin de Carrera se usa para generar una señal con un espectrograma de suficiente resolución entre los 0 y 8500 Hz para observar la respuesta del banco de filtros paso-banda.

En la siguiente Figura se observa el menú de configuración de dicho bloque, donde se pueden especificar parámetros como vectores definidos por la sintaxis de MATLAB.

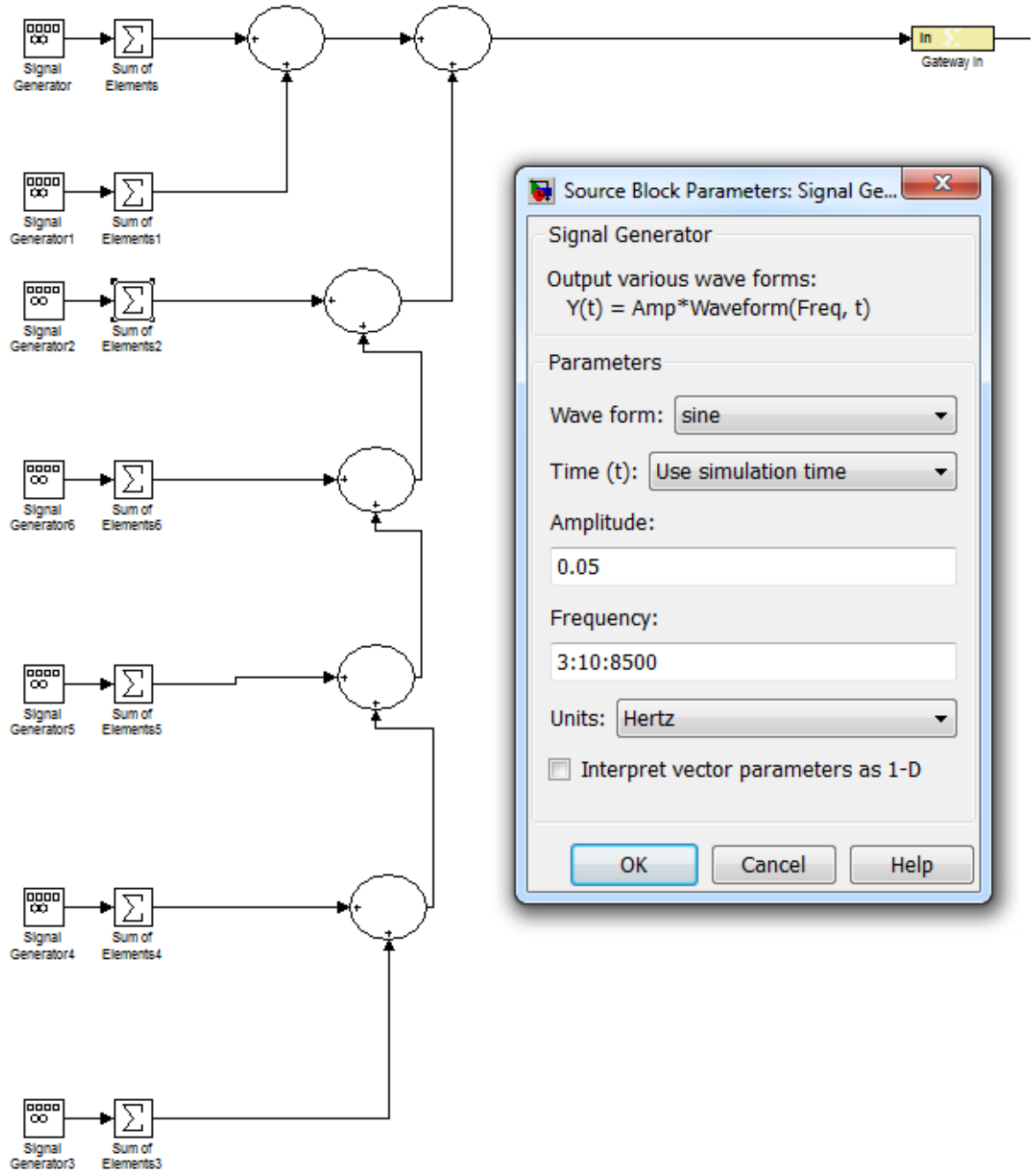


Figura A.8: Signal Generator Menu

A.2.2. Sum of elements

Cuando generamos señales con un bloque Signal Generator mediante la introducción de un vector en el campo de amplitudes o frecuencias es necesario la inclusión posterior de este bloque a fin de obtener un sumatorio de estas señales en cada intervalo temporal.

De no incluir este bloque, se genera un bus de datos a la salida de Signal Generator. Este bus es de la forma 1xN donde N será la longitud del vector amplitud o frecuencia (será el número de señales generadas por este bloque).

En la siguiente figura se muestra el error producido al no incluir este bloque.

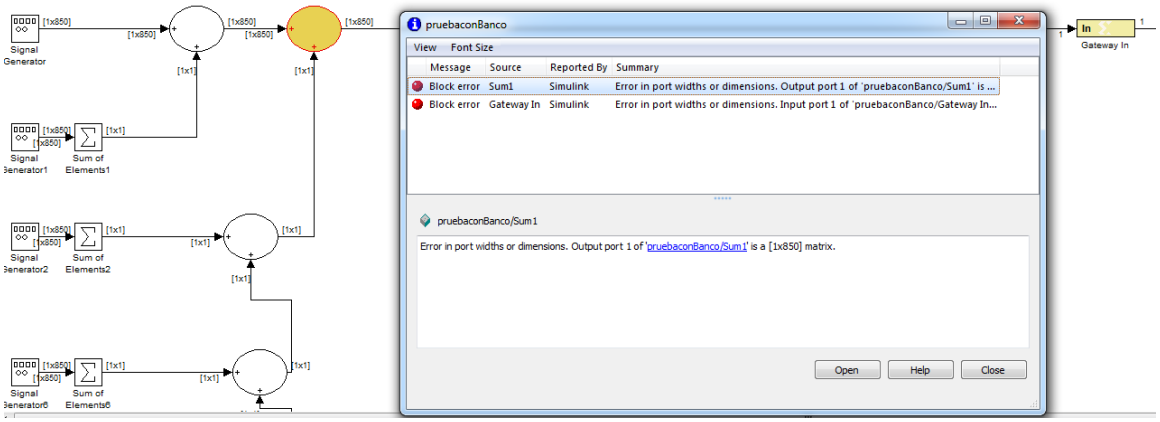


Figura A.9: Error al no incluir el bloque Signal Generator

A.2.3. From Multimedia File

Este bloque nos permite introducir un fichero multimedia con distintos formatos (avi, wav...) en nuestro diseño.

La siguiente figura muestra el bloque y su menú de configuración. En la caja del bloque aparecen la frecuencia de muestreo, el numero de bits de codificación y si el audio es ”mono” o ”stereo”.

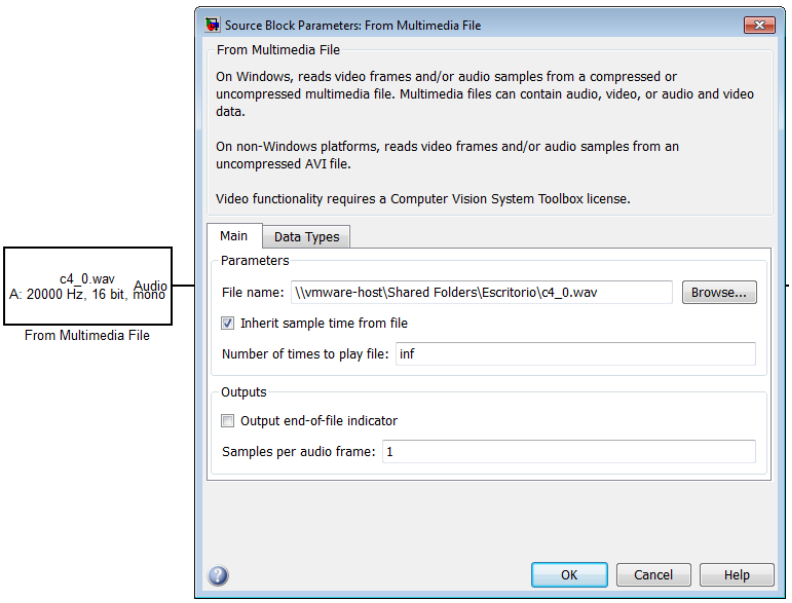


Figura A.10: Configuración de un fichero multimedia

A.2.4. MATLAB Fuction

Aunque en un principio puede establecerse un paralelismo entre este bloque y MCode de System Generator porque ambos tienen la capacidad de contener en su descripción código fuente embebido, no es así. Este bloque permite incluir funciones con lenguaje de alto nivel y permite usar operadores con un índice de complejidad más elevado.

Gracias a esto podemos realizar la cuantificación no lineal necesaria para la descompresión en la fase de reconstrucción. En el código del AnexoB (Referencia) observamos como de forma simple se lleva a cabo dicha operación.

Antes de usar este bloque es necesario que el usuario instale un compilador de C++, el cual no está incluido por defecto.

En este Proyecto Fin de Carrera se ha usado la versión r2011a de MATLAB y el compilador usado fue Visual C++ 2010. Antes de instalar dicho compilador se instaló el SDK 7.1 (Software Development Kit) de Microsoft.

Mathworks proporciona la información concerniente a este tema en su web, dependiendo de la versión de MATLAB usada se tendrá un conjunto de compiladores y soluciones distintas.

Una vez instalado el compilador, deberá ser seleccionado mediante la consola de MATLAB con el comando "mex"setup"

A.2.5. To Multimedia File

Este bloque nos permite almacenar en el ordenador un fichero multimedia a la salida del sistema. Esto facilita el uso de otros programas de análisis independientes de MATLAB como Praat, usado para el análisis de las formantes de la señal de audio reconstruida.

A.2.6. Display

Se trata de una herramienta muy simple que proporciona Simulink para la visualización de valores numéricos en aritmética flotante. En este Proyecto Fin de Carrera se ha usado para comprobar la cuantificación no lineal en el descompresor.

A.2.7. From Workspace

Cuando las variables a usar se encuentran en el área de trabajo de MATLAB, Simulink nos ofrece un bloque capaz de extraer dichas variables a nuestro esquemático. Este bloque se ha usado en la parte de reconstrucción de la señal procesada por el DSP.

En la siguiente figura se pueden observar los bloques Display y From Workspace usados en la reconstrucción

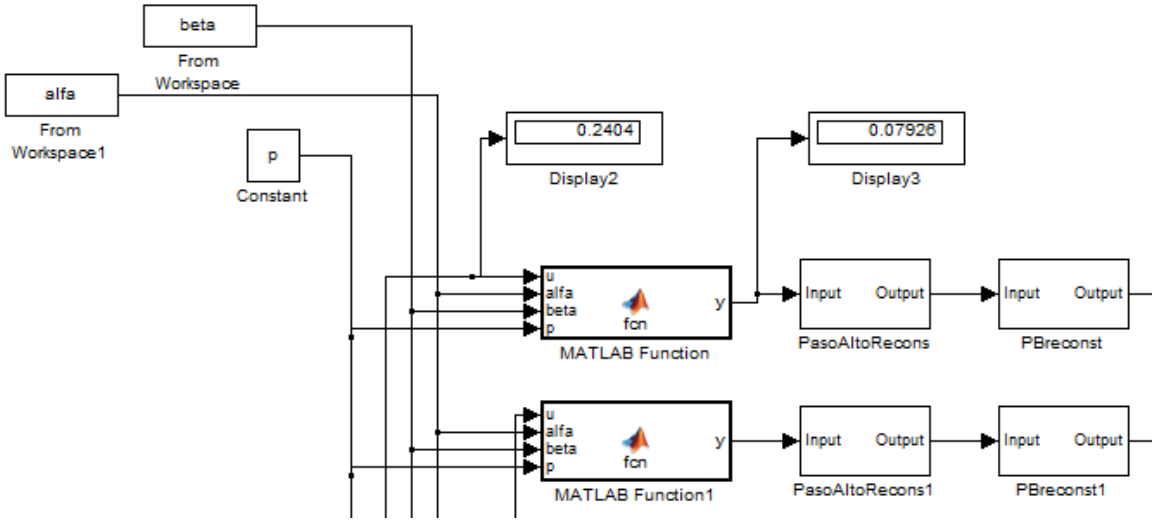


Figura A.11: Bloques Display y From Workspace

A.2.8. Scope

Se trata de una herramienta de visualización de la señal, en aritmética flotante, en el dominio temporal. Este bloque presenta serias limitaciones ya que la visualización de la señal se reinicia cuando modificamos ciertas características del mismo cuando la simulación está en curso. A pesar de estas características sirve para realizar ciertas comprobaciones visuales sobre la simulación.

Principalmente se ha usado para observar el proceso de rectificación de la señal y para observar si, con la tasa de funcionamiento del reloj del sistema, el sistema era capaz de seguir las variaciones de la señal de entrada. También se usó para observar como los bloques Máximo y Mínimo alcanzaban un estado estacionario en el proceso de detección.

A.2.9. Spectrum Scope

Es una herramienta de visualización del espectrograma de una señal en aritmética flotante. Internamente hace la FFT de dicha señal y visualiza la densidad espectral de potencia o el valor cuadrático medio de la señal.

Es necesario configurar este bloque para poder visualizar la señal correctamente. En este caso se ha elegido un buffer de 4096 bits (debe ser potencia de dos) y enventanado Hanning.

La Imagen .ª"muestra la ventana de configuración de propiedades del bloque Spectrum Scope. La Imagen "b"muestra la configuración de los colores de las curvas de representación.

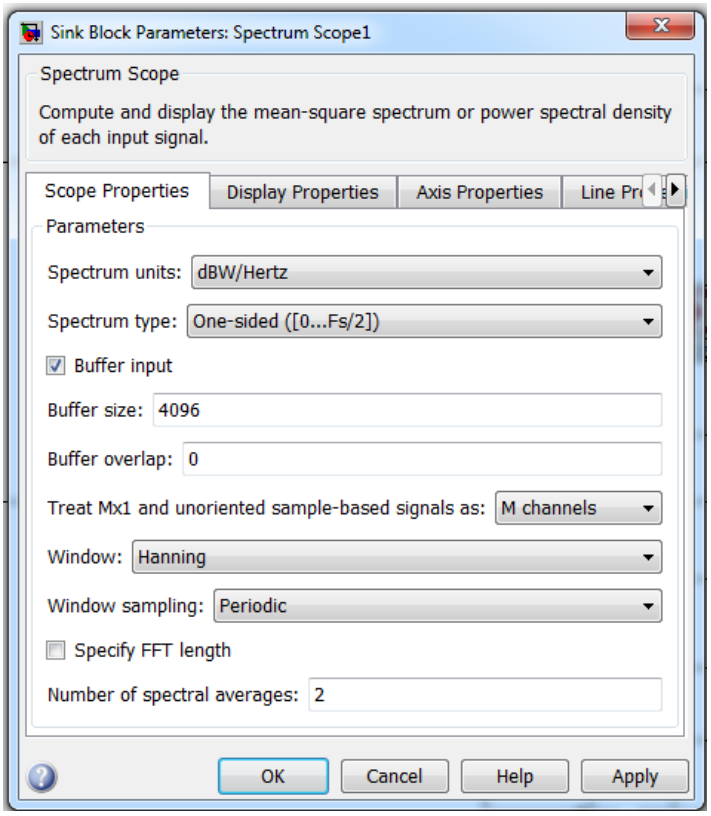


Figura A.12: Imagen a

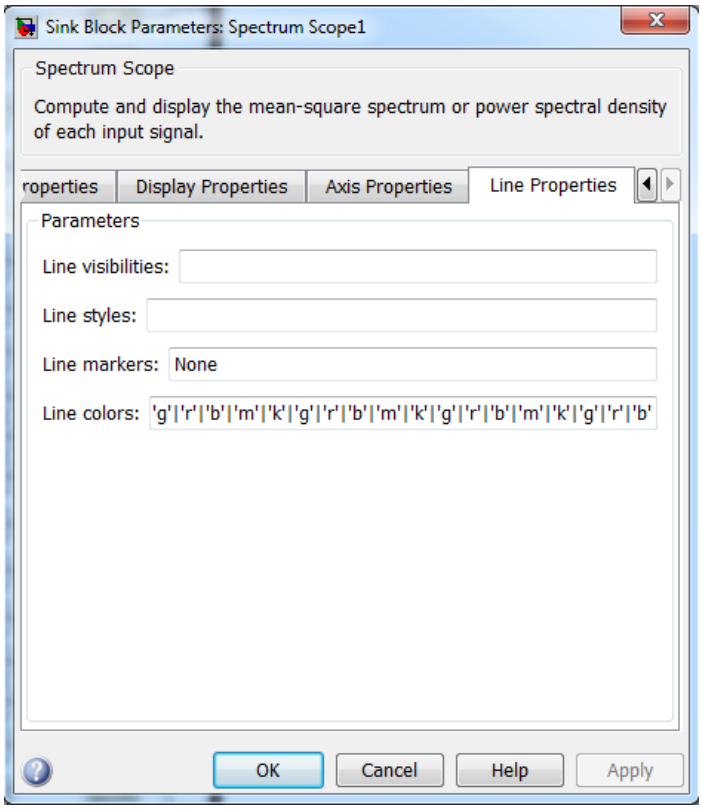


Figura A.13: Imagen b

Anexo B

Anexo 2

CÓDIGO FUENTE MATLAB:

(1) Código fuente del bloque MCode para la detección de banda de linealización dentro del Compresor:

```
   function z = xcuantif(x)
   a=0;
%cuantificación lineal, detecta la banda de linealización del
eje de %abscisas
   for i=0:N-1

   if ((xfix(xlUnsigned,32,16,N *i)<x)(x<xfix(xlUnsigned,32,16,
N*(i+1))))


   a=i;

   end
end
   z=a;
end
```

$N = 2^n$, debe ser declarado en el workspace de MATLAB, ya que MCode no admite la operación potencial en su implementación. La función xfit es imprescindible ya que estamos implementando un sistema que trabaja en punto fijo y no en coma flotante.

(2) El siguiente Código muestra la generación del eje de abscisas (XK) y del eje de ordenadas (Yk) para cada uno de los distintos valores del parámetro de compresión p:

```
   XK(1)=0

   for i=2:2 ^n+1
```

```
   XK(i)=XK(i-1)+1/(2^n)
end

   for k=1:15

   for j=1:2^n+1 Yk(j+(2^n+1)*(k-1))=XK(j)^(k/16);

   end

   end
```

(3) A continuación se muestra el código MATLAB que genera los vectores de coeficientes alfa y beta a partir de los vectores XK e YK. El vector XK contiene los valores discretos del eje de abscisas e YK contiene los valores calculados para el eje de ordenadas de la forma $YK = XK^P$ donde P es un vector con los valores del parámetro p.

```
   clear AUX;

   clear alfa1;

   clear beta1;


   for i=2:1:2^n+1


   alfa1(((2^n)+1)*j+i-1)= (Yk(((2^n)+1)*j+i)-Yk(((2^n)+1)*j+i-1))
/(XK(i)-XK(i-1));

        end
end
   AUX= find((alfa1)~=0)

   alfa=alfa1(AUX);

   clear alfa1;
   for j=0:1:14


   for i=2:1:2^n+1

   beta1(((2^n)+1)*j+i-1)= (XK(i)*Yk(((2^n)+1)*j+i-1)-XK(i-1)
```

```
*Yk(((2^n)+1)*j+i))/(XK(i)-XK(i-1));


   end

   end
   beta=beta1(AUX);
```

(4) Para el cálculo de la densidad espectral de potencia el código MATLAB usado es el siguiente:

```
   x=simout.signals.values
y=fft(x,1024)
f=(22000/1024)*(0:1024/2-1)
Pyy=y.*conj(y)/1024
plot(f,Pyy(1:1024/2),'r')
   hold on
```

(5) El código fuente usado en el bloque del descompresor para llevar a cabo la cuantificación no lineal y la descompresión es el siguiente:

```
   function y = fcn(u,alfa,beta,p)
a=1;
for i=0:N-1
%cuantificación no lineal, detecta la banda del eje de ordenadas


   if ((i/N)^((p+1)/15)<u)   (u<((i+1)/N)^((p+1)/15))

   %revertimos la operación de linealización para calcular
el valor%original de X
a=(u-beta((p+1)*N+i))/alfa((p+1)*N+i)


   end
end
y = a;
```

Siendo N el número de bandas de cuantificación.

Anexo C

Anexo 3

Tabla C.1: Media y desv.típica de las formantes vocálicas en Hz. según diversos autores

		[1]		[2]		[3]		[4]		[5]	
	Vocal	Media	Sd	Media	sd	Media	sd	Media	sd	Media	sd
	[i]	278	39	260	40	313	29	345	31	338	31
	[e]	465	44	411	59	457	40	432	33	436	34
F1	[a]	733	57	636	39	699	83	603	48	594	42
	[o]	510	58	469	87	495	56	477	46	499	41
	[u]	306	61	272	51	349	38	-	-	362	33
	[i]	2254	159	2291	130	2200	153	2089	161	2071	152
	[e]	1972	123	2019	113	1926	117	1764	193	1732	135
F2	[a]	1203	89	1175	74	1471	84	1375	149	1375	109
	[o]	909	62	862	90	1070	114	1062	190	1166	107
	[u]	752	127	671	66	877	128			983	99

Valores medios y desviaciones típicas para F1 y F2 de las vocales en los estudios de [1] M. Rosique, J.L.Ramón, M. Canteras y L. Rosique (1979), [2] Quilis y Esgueva (1983), [3]Martínez Celdrán (1995), [4] Albalá M.J., Battaner E., Carranza M., Gil J., y otros (2008) , [5] Juan Julián Jiménez Gómez

Tabla C.2: Media y desv.típica de las formantes vocálicas en Hz. obtenidas con mis muestras

	Vocal	Media	sd
F1	[i]	359,72	48,34
	[a]	679,65	43,19
	[o]	564,72	88,19
F2	[i]	2085,90	271,03
	[a]	1431,27	58,91
	[o]	1217,07	88,91

Media y desviación típica de las primeras formantes vocálicas teniendo en cuenta a todos los autores mencionados. Se incluyeron también mis muestras

Media i	306,80
sd i	34,30
Media e	440,20
sd e	43,04
Media a	653,00
sd a	56,08
Media o	490,00
sd o	59,78
Media u	322,25
sd u	47,05

Tabla C.3: Media y desviación típica de las segundas formantes vocálicas teniendo en cuenta a todos los autores mencionados. Se incluyeron también mis muestras

Media i	2181,00
sd i	151,40
Media e	1882,60
sd e	139,33
Media a	1319,80
sd a	104,44
Media o	1013,80
sd o	120,41
Media u	820,75
sd u	108,02

Se define la variabilidad relativa de una muestra como $VR = \frac{Desviacin\ tipica}{Media}$. Se utiliza para comparar la variación con respecto a la media que tienen una serie de muestras.

Tabla C.4: Variabilidad relativa de las primeras formantes

Vocal i	11,18 %
Vocal a	8,59 %
Vocal o	12,20 %

Como puede observarse los primeros formantes de la vocal 'a' quedan mejor representados por la media que en el caso de las vocales 'i' y 'o'

Tabla C.5: Variabilidad relativa de las segundas formantes

Vocal i	6,94 %
Vocal a	7,91 %
Vocal o	11,88 %

Como puede observarse los segundos formantes de la vocal 'i' quedan mejor representados por la media que en el caso de las vocales 'a' y 'o'

Tabla C.6: Muestra de los formantes de la vocal 'a' sin FPGA

F1	F2
691,04	1430,81
674,88	1440,03
659,93	1445,37
629,52	1394,67
619,10	1438,34
599,82	1425,49
711,08	1457,14
668,57	1489,19
661,47	1502,45
642,94	1504,50
641,80	1440,41
655,15	1503,82
723,00	1443,47
729,73	1488,19
727,37	1451,77
615,24	1426,66
609,01	1386,49
623,36	1521,49
701,44	1438,80
688,35	1466,67
686,40	1471,61
691,00	1460,36
686,53	1427,07
704,27	1410,46
658,25	1411,13
663,88	1464,67
672,96	1410,68
657,36	1346,76
700,91	1390,51
662,19	1384,51
741,66	1379,21
734,78	1411,73
715,34	1392,70
676,89	1368,83
674,48	1254,33
700,24	1320,61
658,25	1411,13
663,88	1464,67
672,96	1410,68
657,36	1346,76
700,91	1390,51
662,19	1384,51
741,66	1379,21
734,78	1411,73
715,34	1392,70
676,89	1368,83

Tabla C.7: Muestra de los formantes de la vocal 'a' sin FPGA Continuación 1

F1	F2
638,94	1482,24
676,89	1486,70
604,37	1551,86
693,23	1434,12
704,79	1408,60
700,45	1487,15
668,25	1570,47
683,68	1514,90
664,34	1521,61
725,33	1500,03
699,81	1420,20
693,05	1474,13
717,46	1499,87
772,28	1511,09
714,97	1505,20
631,75	1407,07
639,55	1417,51
650,86	1376,60
779,60	1472,47
737,76	1544,77
787,08	1525,85
696,64	1485,52
737,75	1481,49
727,47	1438,78
740,72	1478,43
755,34	1493,01
724,38	1518,95
682,34	1473,77
684,36	1425,50
710,34	1428,75
689,83	1460,56
699,22	1498,28
720,14	1451,85
712,86	1445,16
700,58	1456,86
691,54	1443,93
612,41	1431,41
638,12	1405,04
633,08	1514,91
611,93	1386,76
626,31	1400,43
638,84	1358,49
614,49	1394,41
627,33	1352,22
623,09	1398,09

Tabla C.8: Muestra de los formantes de la vocal 'a' sin FPGA Continuación 2

F1	F2
668,43	1368,95
672,45	1375,91
678,58	1353,67
653,85	1427,46
655,13	1388,44
809,03	1431,37
721,01	1448,45
653,23	1396,45
610,39	1411,41
678,41	1449,56
651,81	1421,21
625,97	1392,19
641,02	1354,62
622,94	1353,36
617,04	1332,94

Tabla C.9: Media y desviación típica de la muestra vocal 'a' sin FPGA

Media F1	679,65
Media F2	1431,27
sd F1	43,19
sd F2	58,91

Tabla C.10: Muestra de los formantes de la vocal 'i' sin FPGA

F1	F2
341,41	2244,73
301,78	2226,73
312,18	2186,70
418,23	2232,84
388,35	2254,52
384,43	2364,49
227,25	2221,95
317,38	2253,80
361,96	2180,24
431,27	2180,60
530,33	2238,80
482,67	2278,52
393,82	2099,88
329,64	2246,25
363,21	2226,52
393,85	2010,61
387,29	2103,48
394,55	2136,26
331,86	2178,08
320,07	2131,88
334,46	2123,55
423,16	2120,66
404,77	2224,40
441,06	2134,32
314,55	2110,89
316,07	2109,95
331,98	2077,90
341,49	1930,58
324,63	1960,46
375,97	1972,11
327,12	2061,31
306,71	2085,50
307,94	2159,20
345,85	1887,11
326,45	1254,66
396,09	2038,20
324,32	1713,21
306,32	1484,06
366,72	2296,97
348,44	2071,11
382,16	2242,52
290,38	1877,01
330,83	1799,35
330,52	2144,59
339,19	1693,14
326,17	1965,00

Tabla C.11: Muestra de los formantes de la vocal 'i' sin FPGA continuación 1

F1	F2
344,38	1990,68
303,26	1918,34
378,59	2206,34
340,55	569,79
296,10	1628,88
360,03	2560,05
395,94	1214,47
398,17	1355,53
345,56	2056,21
314,09	2347,87
311,38	2106,99
418,44	2476,97
380,79	2711,78
363,06	2620,05
361,68	2070,02
376,52	2094,96
326,11	2141,95
389,67	2123,09
378,84	2084,22
391,02	2011,37
349,19	2118,16
349,36	2161,29
300,24	2267,26
364,48	2084,02
370,85	2124,90
442,94	2257,16
319,68	2130,28
313,28	2143,63
304,19	2009,11
411,54	2228,52
393,05	2151,07
378,93	2172,82
304,65	2311,29
311,10	2247,57
305,51	2220,98
353,66	2122,63
363,63	2215,24
209,43	2811,84
328,69	2019,45
325,00	2037,90
315,03	2113,04
454,61	1965,80
399,53	2036,13
415,42	2099,35
332,92	2087,77
326,72	2102,82
346,73	2095,32

Tabla C.12: Muestra de los formantes de la vocal 'i' sin FPGA continuación 2

F1	F2
400,47	2015,09
437,20	2165,74
399,29	2084,55
390,77	2027,67
349,58	2112,88
372,90	2052,11
396,07	2061,03
399,98	2041,69
373,13	2045,57
379,62	2178,28
394,80	2114,50
374,46	2045,54
422,89	2084,63
430,59	2030,05
388,19	2058,60

Tabla C.13: Media y desviación típica de la muestra vocal 'i' sin FPGA

Media F1	359,72
Media F2	2085,90
sd F1	48,34
sd F2	271,03

Tabla C.14: Muestra de los formantes de la vocal 'o' sin FPGA

F1	F2
558,42	1152,16
552,38	1166,25
593,62	1114,78
561,63	1204,16
575,67	1222,34
611,37	1290,69
627,99	1162,88
637,66	1201,70
562,10	1162,79
653,87	1215,35
652,02	1250,41
634,80	1207,93
717,54	1289,59
681,13	1179,42
648,97	1208,01
635,04	1165,38
586,38	1159,92
589,33	1215,80
701,05	1134,97
657,09	1195,27
704,71	1242,86
715,93	1404,44
678,56	1304,41
901,10	1075,33
488,94	1487,14
457,04	1386,59
472,46	1344,01
493,97	1314,20
518,00	1213,52
488,33	1215,22
433,20	1323,29
450,92	1323,44
438,27	1278,35
482,33	1237,65
503,12	1140,13
590,83	1280,61
412,10	1160,06
482,59	1149,20
473,86	1143,41
626,18	1107,50
646,01	1126,18
648,15	1031,04
507,98	1158,10
549,72	1114,95
494,63	1169,23
648,00	1132,86

Tabla C.15: Muestra de los formantes de la vocal 'o' sin FPGA continuación 1

F1	F2
925,98	1362,52
722,63	1236,45
466,50	1116,40
517,71	1181,37
502,36	1119,90
552,55	1109,00
558,49	1105,15
563,50	1071,46
471,28	1136,94
511,20	1139,09
533,29	1159,52
576,20	1246,25
594,87	1424,23
562,91	1169,91
605,94	1331,45
590,17	1440,20
566,94	1298,37
570,37	1445,89
517,26	1327,93
525,77	1301,41
566,36	1270,25
513,36	1284,00
533,06	1277,96
639,95	1229,46
600,78	1247,52
564,96	1286,01
540,77	1255,14
512,85	1272,26
586,91	1362,83
506,58	1261,73
543,68	1224,45
506,95	1287,86
520,93	1300,34
518,14	1221,61
503,42	1263,82
646,77	1152,14
624,85	1309,58
632,71	1249,06
455,44	1230,51
410,17	1206,15
421,42	1259,35
485,81	1237,59
510,58	1208,26
506,59	1131,97
511,85	1304,43
470,02	1173,42
547,70	1136,41

Tabla C.16: Muestra de los formantes de la vocal 'o' sin FPGA continuación 2

601,65	1210,80
581,18	1171,08
588,95	1105,05
485,15	1111,66
469,79	1138,35
413,84	1098,87
619,67	1079,66
600,54	1161,54
622,41	1195,68
491,66	1115,01
517,60	1171,20
530,67	1153,72
641,46	1203,48
625,65	1169,36
636,52	1226,65

Tabla C.17: Media y desviación típica de la muestra vocal 'o' sin FPGA

Media F1	564,72
Media F2	1217,07
sd F1	88,2882944
Sd F2	88,9081937

Tabla C.18: Muestra de las formantes de la vocal 'a' con FPGA

	F1	F2
	749,23	1570,20
	696,80	1546,19
	768,53	1539,65
	713,76	1571,08
	766,24	1560,17
	832,97	1563,85
	728,97	1526,87
	689,74	1543,42
	684,73	1542,07
	630,03	1532,18
	794,40	1506,52
	773,10	1537,55
	633,30	1619,53
	814,54	1599,23
	714,91	1588,10
	758,77	1549,15
	675,04	1566,07
	662,13	1551,30
	701,96	1605,08
	706,76	1551,71
	807,67	1548,00
	793,71	1588,78
	825,71	1525,92
	821,05	1616,35
	767,33	1567,37
	737,48	1558,04
	729,00	1581,26
	588,42	1530,81
	715,05	1495,95
	681,17	1518,39
	644,49	1501,39
	785,38	1554,99
	710,73	1543,12
	739,71	1549,55
	702,75	1562,77
Media	729,87	1554,65
Sd	60,52	29,83

Tabla C.19: Muestra de las formantes de la vocal 'i' con FPGA

	F1	F2
	414,22	2248,93
	398,33	2246,43
	194,99	2239,66
	445,03	2265,27
	558,35	2169,77
	437,73	2254,95
	463,10	2142,72
	496,50	2211,08
	413,74	2099,28
	428,43	2206,45
	511,81	2219,40
	601,30	2214,39
	443,81	1903,01
	425,29	1944,84
	552,83	2223,42
	392,45	1759,13
	499,18	2199,47
	461,85	1950,59
	560,56	2165,65
	396,93	2334,00
	464,91	2214,73
	453,74	2236,33
	459,57	2205,28
	479,55	2249,18
	510,13	2304,26
	466,62	2280,10
	596,77	2283,71
	439,27	2254,07
	486,57	2162,08
	460,42	2271,15
	431,07	2157,71
	551,69	2118,61
	572,65	2289,65
	517,32	2242,21
	474,10	2247,62
	F1	F2
Media	470,31	2186,15
Sd	74,45	122,42

Tabla C.20: Muestra de las formantes de la vocal ’0’ con FPGA

	F1	F2
	698,73	1368,24
	608,71	1277,29
	905,90	1335,29
	847,32	1347,12
	817,57	1234,55
	655,25	1224,60
	526,99	1612,21
	556,89	1756,39
	530,82	1415,62
	512,36	1377,65
	544,93	1624,16
	594,62	1212,23
	561,13	1256,09
	523,96	1396,71
	541,26	1285,28
	561,53	1303,16
	674,88	1338,27
	626,09	1370,73
	589,23	1533,51
	595,80	1283,95
	303,88	1277,10
	581,46	1439,24
	561,66	1464,46
	548,38	1389,26
	640,66	1351,29
	602,47	1255,95
	449,64	1285,68
	587,89	1313,38
	533,57	1339,03
	533,57	1339,03
	742,50	1066,85
	559,44	1268,70
	515,90	1269,04
	595,78	1391,97
	429,70	1224,50
	531,70	992,70
	F1	F2
Media	588,67	1339,48
Sd	112,10	141,52

Tabla C.21: Comparativa 1 sin FPGA

a-F1	a-F2	i-F1	i-F2	o-F1	o-F2
691,04	1430,81	341,41	2244,73	558,42	1152,16
674,88	1440,03	301,78	2226,73	552,38	1166,25
659,93	1445,37	312,18	2186,70	593,62	1114,78
629,52	1394,67	418,23	2232,84	561,63	1204,16
619,10	1438,34	388,35	2254,52	575,67	1222,34
599,82	1425,49	384,43	2364,49	611,37	1290,69
711,08	1457,14	227,25	2221,95	627,99	1162,88
668,57	1489,19	317,38	2253,80	637,66	1201,70
661,47	1502,45	361,96	2180,24	562,10	1162,79
642,94	1504,50	431,27	2180,60	653,87	1215,35
641,80	1440,41	530,33	2238,80	652,02	1250,41
655,15	1503,82	482,67	2278,52	634,80	1207,93
723,00	1443,47	393,82	2099,88	717,54	1289,59
729,73	1488,19	329,64	2246,25	681,13	1179,42
727,37	1451,77	363,21	2226,52	648,97	1208,01
615,24	1426,66	393,85	2010,61	635,04	1165,38
609,01	1386,49	387,29	2103,48	586,38	1159,92
623,36	1521,49	394,55	2136,26	589,33	1215,80
701,44	1438,80	331,86	2178,08	701,05	1134,97
688,35	1466,67	320,07	2131,88	657,09	1195,27
686,40	1471,61	334,46	2123,55	704,71	1242,86
691,00	1460,36	423,16	2120,66	715,93	1404,44
686,53	1427,07	404,77	2224,40	678,56	1304,41
704,27	1410,46	441,06	2134,32	901,10	1075,33
658,25	1411,13	314,55	2110,89	488,94	1487,14
663,88	1464,67	316,07	2109,95	457,04	1386,59
672,96	1410,68	331,98	2077,90	472,46	1344,01
657,36	1346,76	341,49	1930,58	493,97	1314,20
700,91	1390,51	324,63	1960,46	518,00	1213,52
662,19	1384,51	375,97	1972,11	488,33	1215,22
741,66	1379,21	327,12	2061,31	433,20	1323,29
734,78	1411,73	306,71	2085,50	450,92	1323,44
715,34	1392,70	307,94	2159,20	438,27	1278,35
676,89	1368,83	345,85	1887,11	482,33	1237,65
674,48	1254,33	326,45	1254,66	503,12	1140,13
700,24	1320,61	396,09	2038,20	590,83	1280,61
658,25	1411,13	324,32	1713,21	412,10	1160,06
663,88	1464,67	306,32	1484,06	482,59	1149,20
672,96	1410,68	366,72	2296,97	473,86	1143,41
657,36	1346,76	348,44	2071,11	626,18	1107,50
700,91	1390,51	382,16	2242,52	646,01	1126,18
662,19	1384,51	290,38	1877,01	648,15	1031,04
741,66	1379,21	330,83	1799,35	507,98	1158,10
734,78	1411,73	330,52	2144,59	549,72	1114,95
715,34	1392,70	339,19	1693,14	494,63	1169,23
676,89	1368,83	326,17	1965,00	648,00	1132,86

Tabla C.22: Comparativa 2 sin FPGA

a-F1	a-F2	i-F1	i-F2	o-F1	o-F2
674,48	1254,33	344,38	1990,68	925,98	1362,52
700,24	1320,61	303,26	1918,34	722,63	1236,45
638,94	1482,24	378,59	2206,34	466,50	1116,40
676,89	1486,70	340,55	569,79	517,71	1181,37
604,37	1551,86	296,10	1628,88	502,36	1119,90
693,23	1434,12	360,03	2560,05	552,55	1109,00
704,79	1408,60	395,94	1214,47	558,49	1105,15
700,45	1487,15	398,17	1355,53	563,50	1071,46
668,25	1570,47	345,56	2056,21	471,28	1136,94
683,68	1514,90	314,09	2347,87	511,20	1139,09
664,34	1521,61	311,38	2106,99	533,29	1159,52
725,33	1500,03	418,44	2476,97	576,20	1246,25
699,81	1420,20	380,79	2711,78	594,87	1424,23
693,05	1474,13	363,06	2620,05	562,91	1169,91
717,46	1499,87	361,68	2070,02	605,94	1331,45
772,28	1511,09	376,52	2094,96	590,17	1440,20
714,97	1505,20	326,11	2141,95	566,94	1298,37
631,75	1407,07	389,67	2123,09	570,37	1445,89
639,55	1417,51	378,84	2084,22	517,26	1327,93
650,86	1376,60	391,02	2011,37	525,77	1301,41
779,60	1472,47	349,19	2118,16	566,36	1270,25
737,76	1544,77	349,36	2161,29	513,36	1284,00
787,08	1525,85	300,24	2267,26	533,06	1277,96
696,64	1485,52	364,48	2084,02	639,95	1229,46
737,75	1481,49	370,85	2124,90	600,78	1247,52
727,47	1438,78	442,94	2257,16	564,96	1286,01
740,72	1478,43	319,68	2130,28	540,77	1255,14
755,34	1493,01	313,28	2143,63	512,85	1272,26
724,38	1518,95	304,19	2009,11	586,91	1362,83
682,34	1473,77	411,54	2228,52	506,58	1261,73
684,36	1425,50	393,05	2151,07	543,68	1224,45
710,34	1428,75	378,93	2172,82	506,95	1287,86
689,83	1460,56	304,65	2311,29	520,93	1300,34
699,22	1498,28	311,10	2247,57	518,14	1221,61
720,14	1451,85	305,51	2220,98	503,42	1263,82
712,86	1445,16	353,66	2122,63	646,77	1152,14
700,58	1456,86	363,63	2215,24	624,85	1309,58
691,54	1443,93	209,43	2811,84	632,71	1249,06
612,41	1431,41	328,69	2019,45	455,44	1230,51
638,12	1405,04	325,00	2037,90	410,17	1206,15
633,08	1514,91	315,03	2113,04	421,42	1259,35
611,93	1386,76	454,61	1965,80	485,81	1237,59
626,31	1400,43	399,53	2036,13	510,58	1208,26
638,84	1358,49	415,42	2099,35	506,59	1131,97
614,49	1394,41	332,92	2087,77	511,85	1304,43
627,33	1352,22	326,72	2102,82	470,02	1173,42

Tabla C.23: Comparativa 3 sin FPGA

a-F1	a-F2	i-F1	i-F2	o-F1	o-F2
623,09	1398,09	346,73	2095,32	547,70	1136,41
668,43	1368,95	400,47	2015,09	601,65	1210,80
672,45	1375,91	437,20	2165,74	581,18	1171,08
678,58	1353,67	399,29	2084,55	588,95	1105,05
653,85	1427,46	390,77	2027,67	485,15	1111,66
655,13	1388,44	349,58	2112,88	469,79	1138,35
809,03	1431,37	372,90	2052,11	413,84	1098,87
721,01	1448,45	396,07	2061,03	619,67	1079,66
653,23	1396,45	399,98	2041,69	600,54	1161,54
610,39	1411,41	373,13	2045,57	622,41	1195,68
678,41	1449,56	379,62	2178,28	491,66	1115,01
651,81	1421,21	394,80	2114,50	517,60	1171,20
625,97	1392,19	374,46	2045,54	530,67	1153,72
641,02	1354,62	422,89	2084,63	641,46	1203,48
622,94	1353,36	430,59	2030,05	625,65	1169,36
617,04	1332,94	388,19	2058,60	636,52	1226,65

Tabla C.24: Comparativa desvtip sin FPGA

Vocal sin FPGA	F1	F2
a	43,190	58,910
i	48,340	271,030
o	88,290	88,910

Tabla C.25: Comparativa con FPGA

a-FPGA F1	a-FPGA F2	i-FPGA F1	i-FPGA F2	o-FPGA F1	o-FPGA F2
749,23	1570,20	414,22	2248,93	698,73	1368,24
696,80	1546,19	398,33	2246,43	608,71	1277,29
768,53	1539,65	194,99	2239,66	905,90	1335,29
713,76	1571,08	445,03	2265,27	847,32	1347,12
766,24	1560,17	558,35	2169,77	817,57	1234,55
832,97	1563,85	437,73	2254,95	655,25	1224,60
728,97	1526,87	463,10	2142,72	526,99	1612,21
689,74	1543,42	496,50	2211,08	556,89	1756,39
684,73	1542,07	413,74	2099,28	530,82	1415,62
630,03	1532,18	428,43	2206,45	512,36	1377,65
794,40	1506,52	511,81	2219,40	544,93	1624,16
773,10	1537,55	601,30	2214,39	594,62	1212,23
633,30	1619,53	443,81	1903,01	561,13	1256,09
814,54	1599,23	425,29	1944,84	523,96	1396,71
714,91	1588,10	552,83	2223,42	541,26	1285,28
758,77	1549,15	392,45	1759,13	561,53	1303,16
675,04	1566,07	499,18	2199,47	674,88	1338,27
662,13	1551,30	461,85	1950,59	626,09	1370,73
701,96	1605,08	560,56	2165,65	589,23	1533,51
706,76	1551,71	396,93	2334,00	595,80	1283,95
807,67	1548,00	464,91	2214,73	303,88	1277,10
793,71	1588,78	453,74	2236,33	581,46	1439,24
825,71	1525,92	459,57	2205,28	561,66	1464,46
821,05	1616,35	479,55	2249,18	548,38	1389,26
767,33	1567,37	510,13	2304,26	640,66	1351,29
737,48	1558,04	466,62	2280,10	602,47	1255,95
729,00	1581,26	596,77	2283,71	449,64	1285,68
588,42	1530,81	439,27	2254,07	587,89	1313,38
715,05	1495,95	486,57	2162,08	533,57	1339,03
681,17	1518,39	460,42	2271,15	533,57	1339,03
644,49	1501,39	431,07	2157,71	742,50	1066,85
785,38	1554,99	551,69	2118,61	559,44	1268,70
710,73	1543,12	572,65	2289,65	515,90	1269,04
739,71	1549,55	517,32	2242,21	595,78	1391,97
702,75	1562,77	474,10	2247,62	429,70	1224,50
				531,70	992,70

Tabla C.26: Formantes originales de las cinco vocales

aF1	aF2	eF1	eF2	iF1	iF2	oF1	oF2	uF1	uF2
836,740	1300,715	654,45	1963,90	432,94	2266,19	530,73	1190,64	329,76	706,98
686,355	1232,845	503,99	1849,25	379,71	2250,48	517,33	874,27	316,51	710,73
717,993	1160,398	330,98	2118,59	356,38	2395,23	575,89	939,63	445,43	592,52
734,298	1507,646	490,43	2310,87	290,57	2324,81	692,81	1091,98	414,47	774,92
736,774	1234,181	453,80	2042,30	449,48	2261,13	536,43	914,13	348,71	761,40
855,92	1285,87	454,21	2150,76	265,263	2345,641	436,90	872,76	123,78	805,79
793,33	1455,19	424,55	2187,85	259,346	2571,031	572,78	987,21	380,50	927,79
933,21	1285,35	488,85	2196,75	334,630	2589,400	610,69	932,59	376,37	737,54
743,36	1303,39	465,36	2106,90	289,877	2329,484	585,99	920,39	417,15	669,16
991,94	1387,65	506,97	2417,17	303,910	2494,464	625,65	1065,76	427,47	893,84

Tabla C.27: Formantes de las cinco vocales después de ser procesadas

aFPGA1	aFPGAF2	eFPGA1	eFPGA2	iFPGA1	iFPGA2	oFPGA1	oFPGA2	uFPGA1	uFPGA2
885,43	1298,77	584,36	2279,49	370,08	2195,66	570,03	960,04	584,93	1358,84
810,78	1362,27	519,48	2172,47	400,40	2164,16	813,63	953,68	546,00	1261,00
867,96	1340,89	475,33	2258,45	493,38	2651,50	675,00	1041,00	523,98	1210,28
843,81	1155,11	475,72	2151,99	370,72	2387,01	606,28	1078,46	499,32	1111,26
778,78	1336,10	486,92	2028,97	365,51	2471,11	586,42	862,64	537,00	1294,80
816,58	1506,95	504,68	2113,15	452,77	2268,13	563,67	809,80	533,62	1239,40
803,08	1453,59	538,97	2058,15	506,12	2313,99	659,00	1031,70	504,73	967,32
948,12	1333,80	481,40	2119,86	452,02	2464,60	705,56	1046,87	458,89	1349,90
997,50	1279,22	399,08	2025,22	330,21	2485,72	759,96	993,03	429,24	1272,63
874,18	1487,50	447,76	2226,27	370,21	2436,71	717,52	1105,24	456,28	1301,32

Referencias Bibliográficas

[1] C. Morales Angulo (Servicio de ORL. Hospital Santillana. Torrelavega), J. Gallo-Terán, N. Azuara, J. Rama Quintela (Servicio de ORL. Hospital Marqués de Valdecilla. Santander). Etiología de la hipoacusia severa/profunda bilateral pre/perilocutiva en Cantabria.

[2] Julio Felix Muñiz. Estudio de la correlación existente entre el efecto supresor contralateral y la fatiga auditiva mediante otoemisiones acústicas transitorias (capítulo 3, sección 5, subsección 3). Universitat de Valencia - Servei de Publicacions.

[3] Sancho Serrano E, Escorial Sanz O, Sebastián Cortés JM, Rivas Rodriguez P, Jiménez Vergara M, Vallés Varela H. Malformación congénita del oído interno. Displasia de Mondini. Servicio ORL.: Hospital Clínico Universitario "Lozano Blesa" Zaragoza.

[4] National Institutes of Health, U.S. Departmentof Health and Human Services.: NIDCD Fact Sheet Cochlear Implants, National Institute on Deafness and Other Communication Disorders.

[5] Memoria de Tesis Doctoral presentada por Francisco de Borja Rodríguez Ortiz Dirigida por Vicente López Martínez Prof. Titular de Ciencias de la Computación e Inteligencia Artificial en la Escuela Técnica Superior de Informática.: Procesos de Estabilización,Sincronización y Aprendizaje en Redes Neuronales Estocásticas (Capítulo I). .: Universidad Autónoma de Madrid.

[6] Zwolan TA, Kileny PR, Smith S, Waltzman S, Chute P, Domico E, Firszt J, Hodges A, Mills D, Whearty M, Osberger MJ, Fisher L. Comparison of continuous interleaved sampling and simultaneous analog stimulation speech processing strategies in newly implanted adults with a Clarion 1.2 cochlear implant. University of Michigan, Ann Arbor, Michigan, USA.

[7] Alejandro Osses Vecchi, Víctor Espinoza Catalán. Análisis comparativo entre modelos de sonoridad: propuesta de un modelo de ponderación dinámica. Departamento de Sonido y Acústica, Universidad Tecnológica de Chile, INACAP. Brown Norte 290, Ñuñoa, Santiago, Chile. Centro Tecnológico, Facultad de Artes, Universidad de Chile. Compañía 1264, Santiago,Chile.

[8] Debalina Ghosh, Depanwita Sarkar Debnath, Saikat Bose. A comparative study of performance of fpga based mel filter bank bark filter bank. International Journal of Artificial Intelligence Applications (IJAIA), Vol.3, No.3, May 2012

[9] Bernard Katz. Mechanism of Synaptic Transmission. Reviews of Modern Physics, Volume 31, number 2 (April 1959)

[10] Manuel Rodriguez Valido. Martín Gutiérrez Castañeda. Eduardo Magdaleno Castello. David Hernández Expósito. Fernando Pérez Nava. Lucas Guerrero Vidal. Metodología de diseño en FPGA usando Xilinx System Generator. Universidad de La Laguna, ULL La Laguna, Tenerife, España.

[11] Bruce H. Edwards (Department of Mathematics - University of Florida), Rafael A. Díaz (Pasadena Independent School District - Sam Rayburn, High School Department Mathematics). ¿Cómo Multiplican y dividen las calculadoras?

[12] Juan Julián Jiménez Gómez (e-mail: juanjulianjimenez@gmail.com). Estructura formántica y campo de dispersión de las vocales del español en telefonía móvil (Estudios Fónicos / Cuadernos de Trabajo 1 (2011), 39-58)

[13] David S. Moore. The Basic Practice of Statistics, 2^{nd} Edition (W.H. FREEMAN AND COMPANY, New York and Basingtoke)

[14] http://www-01.ibm.com/software/es/analytics/spss/products/statistics/